```
**********************************************************
    3858 Thu Jul 11 12:26:10 2013
new/usr/src/head/iso/locale_iso.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License, Version 1.0 only
 * (the "License").  You may not use this file except in compliance
 * with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*      Copyright (c) 1988 AT&T */
/*        All Rights Reserved   */


/*
 * An application should not include this header directly.  Instead it
 * should be included only through the inclusion of other Sun headers.
 *
 * The contents of this header is limited to identifiers specified in the
 * C Standard.  Any new identifiers specified in future amendments to the
 * C Standard must be placed in this header.  If these new identifiers
 * are required to also be in the C++ Standard "std" namespace, then for
 * anything other than macro definitions, corresponding "using" directives
 * must also be added to <locale.h>.
 */

#ifndef _ISO_LOCALE_ISO_H
#define _ISO_LOCALE_ISO_H

#pragma ident   "%Z%%M% %I%     %E% SMI"

#include <sys/feature_tests.h>

#ifdef __cplusplus
extern "C" {
#endif

#if __cplusplus >= 199711L
namespace std {
#endif

struct lconv {
        char *decimal_point;
        char *thousands_sep;
        char *grouping;
```



```
        char *int_curr_symbol;
        char *currency_symbol;
        char *mon_decimal_point;
        char *mon_thousands_sep;
        char *mon_grouping;
        char *positive_sign;
        char *negative_sign;
        char int_frac_digits;
        char frac_digits;
        char p_cs_precedes;
        char p_sep_by_space;
        char n_cs_precedes;
        char n_sep_by_space;
        char p_sign_posn;
        char n_sign_posn;

/*
 * New in IEEE Std 1003.1-2001 for alignment with the ISO/IEC 9899:1999
 * standard.  Namespace and binary compatibility dictate that visibility
 * of these new members be limited.  Visibility is limited to a strictly
 * conforming ANSI C environment (-Xc) or if _LCONV_C99 is defined.
 */
#if (defined(_STRICT_STDC) && defined(_STDC_C99)) || defined(_LCONV_C99)
        char int_p_cs_precedes;
        char int_p_sep_by_space;
        char int_n_cs_precedes;
        char int_n_sep_by_space;
        char int_p_sign_posn;
        char int_n_sign_posn;
#endif
};

#define LC_CTYPE        0
#define LC_NUMERIC      1
#define LC_TIME         2
#define LC_COLLATE      3
#define LC_MONETARY     4
#define LC_MESSAGES     5
#define LC_ALL          6

#ifndef NULL
#if defined(_LP64)
#define NULL    0L
#else
#define NULL    0
#endif
#endif

#if     defined(__STDC__)
extern char     *setlocale(int, const char *);
extern struct lconv *localeconv(void);
#else
extern char   *setlocale();
extern struct lconv     *localeconv();
#endif

#define LC_COLLATE_MASK         (1<<0)
#define LC_CTYPE_MASK           (1<<1)
#define LC_MESSAGES_MASK        (1<<2)
#define LC_MONETARY_MASK        (1<<3)
#define LC_NUMERIC_MASK         (1<<4)
#define LC_TIME_MASK            (1<<5)
#define LC_ALL_MASK             (LC_COLLATE_MASK | LC_CTYPE_MASK | LC_MESSAGES_M
                                 LC_MONETARY_MASK | LC_NUMERIC_MASK | LC_TIME_MA
#define LC_GLOBAL_LOCALE        ((locale_t)-1)
```

```
126 /* XXX */
127 #ifndef _LOCALE_T_DEFINED
128 #define _LOCALE_T_DEFINED
129 typedef struct  _xlocale *locale_t;
130 #endif

132 locale_t        duplocale(locale_t);
133 int             freelocale(locale_t);
134 locale_t        newlocale(int, const char *, locale_t);
135 const char      *querylocale(int, locale_t);
136 locale_t        uselocale(locale_t);

138 #endif /* ! codereview */
139 #if __cplusplus >= 199711L
140 }
141 #endif /* end of namespace std */

143 #ifdef  __cplusplus
144 }
145 #endif

147 #endif  /* _ISO_LOCALE_ISO_H */
```

```
**********************************************************
    4557 Thu Jul 11 12:26:10 2013
new/usr/src/head/langinfo.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
   1 /*
   2  * CDDL HEADER START
   3  *
   4  * The contents of this file are subject to the terms of the
   5  * Common Development and Distribution License, Version 1.0 only
   6  * (the "License").  You may not use this file except in compliance
   7  * with the License.
   8  *
   9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
  10  * or http://www.opensolaris.org/os/licensing.
  11  * See the License for the specific language governing permissions
  12  * and limitations under the License.
  13  *
  14  * When distributing Covered Code, include this CDDL HEADER in each
  15  * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  16  * If applicable, add the following below this CDDL HEADER, with the
  17  * fields enclosed by brackets "[]" replaced with your own identifying
  18  * information: Portions Copyright [yyyy] [name of copyright owner]
  19  *
  20  * CDDL HEADER END
  21  */
  22 /*
  23  * Copyright 2003 Sun Microsystems, Inc.  All rights reserved.
  24  * Use is subject to license terms.
  25  */

  27 /*      Copyright (c) 1988 AT&T */
  28 /*        All Rights Reserved   */


  31 #ifndef _LANGINFO_H
  32 #define _LANGINFO_H

  34 #pragma ident   "%Z%%M% %I%     %E% SMI"        /* SVr4.0 1.2   */

  34 #include <sys/feature_tests.h>
  35 #include <nl_types.h>

  37 #ifdef  __cplusplus
  38 extern "C" {
  39 #endif

  41 /*
  42  * The seven days of the week in their full beauty
  43  */

  45 #define DAY_1     1     /* sunday */
  46 #define DAY_2     2     /* monday */
  47 #define DAY_3     3     /* tuesday */
  48 #define DAY_4     4     /* wednesday */
  49 #define DAY_5     5     /* thursday */
  50 #define DAY_6     6     /* friday */
  51 #define DAY_7     7     /* saturday */

  53 /*
  54  * The abbreviated seven days of the week
  55  */

  57 #define ABDAY_1   8  /* sun */
  58 #define ABDAY_2   9  /* mon */
  59 #define ABDAY_3   10 /* tue */
```



```
  60 #define ABDAY_4   11 /* wed */
  61 #define ABDAY_5   12 /* thu */
  62 #define ABDAY_6   13 /* fri */
  63 #define ABDAY_7   14 /* sat */

  65 /*
  66  * The full names of the twelve months...
  67  */

  69 #define MON_1     15 /* january */
  70 #define MON_2     16 /* february */
  71 #define MON_3     17 /* march */
  72 #define MON_4     18 /* april */
  73 #define MON_5     19 /* may */
  74 #define MON_6     20 /* june */
  75 #define MON_7     21 /* july */
  76 #define MON_8     22 /* august */
  77 #define MON_9     23 /* september */
  78 #define MON_10    24 /* october */
  79 #define MON_11    25 /* november */
  80 #define MON_12    26 /* december */

  82 /*
  83  * ... and their abbreviated form
  84  */

  86 #define ABMON_1   27 /* jan */
  87 #define ABMON_2   28 /* feb */
  88 #define ABMON_3   29 /* mar */
  89 #define ABMON_4   30 /* apr */
  90 #define ABMON_5   31 /* may */
  91 #define ABMON_6   32 /* jun */
  92 #define ABMON_7   33 /* jul */
  93 #define ABMON_8   34 /* aug */
  94 #define ABMON_9   35 /* sep */
  95 #define ABMON_10  36 /* oct */
  96 #define ABMON_11  37 /* nov */
  97 #define ABMON_12  38 /* dec */

  99 /*
 100  * plus some special strings you might need to know
 101  */

 103 #define RADIXCHAR 39    /* radix character */
 104 #define THOUSEP   40    /* separator for thousand */
 105 /* YESSTR and NOSTR marked as legacy in XPG5 and removed in SUSv3 */
 106 #if !defined(_XPG6) || defined(__EXTENSIONS__)
 107 #define YESSTR    41    /* affirmative response for yes/no queries */
 108 #define NOSTR     42    /* negative response for yes/no queries */
 109 #endif /* !defined(_XPG6) || defined(__EXTENSIONS__ */
 110 #define CRNCYSTR  43    /* currency symbol */

 112 /*
 113  * Default string used to format date and time
 114  *      e.g. Sunday, August 24 21:08:38 MET 1986
 115  */

 117 #define D_T_FMT   44    /* string for formatting date and time */
 118 #define D_FMT     45    /* date format */
 119 #define T_FMT     46    /* time format */
 120 #define AM_STR    47    /* am string */
 121 #define PM_STR    48    /* pm string */

 123 /*
 124  * Additions for XPG4 (XSH4) Compliance
 125  */
```

```
127 #define CODESET         49      /* codeset name */
128 #define T_FMT_AMPM      50      /* am or pm time format string */
129 #define ERA             51      /* era description segments */
130 #define ERA_D_FMT       52      /* era date format string */
131 #define ERA_D_T_FMT     53      /* era date and time format string */
132 #define ERA_T_FMT       54      /* era time format string */
133 #define ALT_DIGITS      55      /* alternative symbols for digits */
134 #define YESEXPR         56      /* affirmative response expression */
135 #define NOEXPR          57      /* negative response expression */
136 #define _DATE_FMT       58      /* strftime format for date(1) */

138 #if defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE)
139 #define MAXSTRMSG       58 /* Maximum number of strings in langinfo */
140 #endif /* defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE) */

142 #ifndef _LOCALE_T_DEFINED
143 #define _LOCALE_T_DEFINED
144 typedef struct  _xlocale *locale_t;
145 #endif

147 #endif /* ! codereview */
148 /*
149  * and the definitions of functions langinfo(3C)
150  */
151 #if defined(__STDC__)
152 char    *nl_langinfo(nl_item);  /* get a string from the database       */
153 char    *nl_langinfo_l(nl_item, locale_t);
154 #endif /* ! codereview */
155 #else
156 char    *nl_langinfo();         /* get a string from the database       */
157 char    *nl_langinfo_l();
158 #endif /* ! codereview */
159 #endif

161 #ifdef  __cplusplus
162 }
163 #endif

165 #endif  /* _LANGINFO_H */
```

```
**********************************************************
   22424 Thu Jul 11 12:26:10 2013
new/usr/src/lib/libc/amd64/Makefile
2964 need POSIX 2008 locale object support
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 2004, 2010, Oracle and/or its affiliates. All rights reserved.
#
# Copyright (c) 2013, OmniTI Computer Consulting, Inc. All rights reserved.
# Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
# Use is subject to license terms.
#

LIBCBASE=       .
LIBCDIR=        $(SRC)/lib/libc
LIBRARY=        libc.a
LIB_PIC=        libc_pic.a
VERS=           .1
CPP=            /usr/lib/cpp
TARGET_ARCH=    amd64

# objects are grouped by source directory

# local objects
STRETS=

CRTOBJS=                        \
        cerror.o

DYNOBJS=

FPOBJS=                         \
        _base_il.o              \
        fpgetmask.o             \
        fpgetround.o            \
        fpsetmask.o             \
        fpsetround.o            \
        fpstart.o

I386FPOBJS=                     \
        _D_cplx_div.o           \
        _D_cplx_div_ix.o        \
        _D_cplx_div_rx.o        \
        _D_cplx_lr_div.o        \
        _D_cplx_lr_div_ix.o     \
        _D_cplx_lr_div_rx.o     \
```



```
        _D_cplx_mul.o           \
        _F_cplx_div.o           \
        _F_cplx_div_ix.o        \
        _F_cplx_div_rx.o        \
        _F_cplx_lr_div.o        \
        _F_cplx_lr_div_ix.o     \
        _F_cplx_lr_div_rx.o     \
        _F_cplx_mul.o           \
        _X_cplx_div.o           \
        _X_cplx_div_ix.o        \
        _X_cplx_div_rx.o        \
        _X_cplx_lr_div.o        \
        _X_cplx_lr_div_ix.o     \
        _X_cplx_lr_div_rx.o     \
        _X_cplx_mul.o

FPASMOBJS=                      \
        __xgetRD.o              \
        _xtoll.o                \
        _xtoull.o               \
        fpcw.o                  \
        fpgetsticky.o           \
        fpsetsticky.o

ATOMICOBJS=                     \
        atomic.o

XATTROBJS=                      \
        xattr_common.o
COMOBJS=                        \
        bcmp.o                  \
        bcopy.o                 \
        bsearch.o               \
        bzero.o                 \
        qsort.o                 \
        strtol.o                \
        strtoul.o               \
        strtoll.o               \
        strtoull.o

GENOBJS=                        \
        _getsp.o                \
        abs.o                   \
        alloca.o                \
        attrat.o                \
        byteorder.o             \
        cuexit.o                \
        ecvt.o                  \
        errlst.o                \
        amd64_data.o            \
        ldivide.o               \
        lock.o                  \
        makectxt.o              \
        memccpy.o               \
        memchr.o                \
        memcmp.o                \
        memcpy.o                \
        memset.o                \
        new_list.o              \
        proc64_id.o             \
        proc64_support.o        \
        setjmp.o                \
        siginfolst.o            \
        siglongjmp.o            \
        strcmp.o                \
        strcpy.o                \
```

```
128         strlen.o                \
129         strncmp.o               \
130         strncpy.o               \
131         strnlen.o               \
132         sync_instruction_memory.o

134 # Preserved solely to ease maintenance of 32-bit and 64-bit library builds
135 # This macro should ALWAYS be empty; native APIs are already 'large file'.
136 COMSYSOBJS64=

138 SYSOBJS64=

140 COMSYSOBJS=                     \
141         __clock_timer.o         \
142         __getloadavg.o          \
143         __rusagesys.o           \
144         __signotify.o           \
145         __sigrt.o               \
146         __time.o                \
147         _lgrp_home_fast.o       \
148         _lgrpsys.o              \
149         _nfssys.o               \
150         _portfs.o               \
151         _pset.o                 \
152         _rpcsys.o               \
153         _sigaction.o            \
154         _so_accept.o            \
155         _so_bind.o              \
156         _so_connect.o           \
157         _so_getpeername.o       \
158         _so_getsockname.o       \
159         _so_getsockopt.o        \
160         _so_listen.o            \
161         _so_recv.o              \
162         _so_recvfrom.o          \
163         _so_recvmsg.o           \
164         _so_send.o              \
165         _so_sendmsg.o           \
166         _so_sendto.o            \
167         _so_setsockopt.o        \
168         _so_shutdown.o          \
169         _so_socket.o            \
170         _so_socketpair.o        \
171         _sockconfig.o           \
172         acct.o                  \
173         acl.o                   \
174         adjtime.o               \
175         alarm.o                 \
176         brk.o                   \
177         chdir.o                 \
178         chroot.o                \
179         cladm.o                 \
180         close.o                 \
181         execve.o                \
182         exit.o                  \
183         facl.o                  \
184         fchdir.o                \
185         fchroot.o               \
186         fdsync.o                \
187         fpathconf.o             \
188         fstatfs.o               \
189         fstatvfs.o              \
190         getcpuid.o              \
191         getdents.o              \
192         getegid.o               \
193         geteuid.o               \
```



```
194         getgid.o                \
195         getgroups.o             \
196         gethrtime.o             \
197         getitimer.o             \
198         getmsg.o                \
199         getpid.o                \
200         getpmsg.o               \
201         getppid.o               \
202         getrlimit.o             \
203         getuid.o                \
204         gtty.o                  \
205         install_utrap.o         \
206         ioctl.o                 \
207         kaio.o                  \
208         kill.o                  \
209         llseek.o                \
210         lseek.o                 \
211         mmapobjsys.o            \
212         memcntl.o               \
213         mincore.o               \
214         mmap.o                  \
215         modctl.o                \
216         mount.o                 \
217         mprotect.o              \
218         munmap.o                \
219         nice.o                  \
220         ntp_adjtime.o           \
221         ntp_gettime.o           \
222         p_online.o              \
223         pathconf.o              \
224         pause.o                 \
225         pcsample.o              \
226         pipe2.o                 \
227         pollsys.o               \
228         pread.o                 \
229         priocntlset.o           \
230         processor_bind.o        \
231         processor_info.o        \
232         profil.o                \
233         putmsg.o                \
234         putpmsg.o               \
235         pwrite.o                \
236         read.o                  \
237         readv.o                 \
238         resolvepath.o           \
239         seteguid.o              \
240         setgid.o                \
241         setgroups.o             \
242         setitimer.o             \
243         setreid.o               \
244         setrlimit.o             \
245         setuid.o                \
246         sigaltstk.o             \
247         sigprocmsk.o            \
248         sigsendset.o            \
249         sigsuspend.o            \
250         statfs.o                \
251         statvfs.o               \
252         stty.o                  \
253         sync.o                  \
254         sysconfig.o             \
255         sysfs.o                 \
256         sysinfo.o               \
257         syslwp.o                \
258         times.o                 \
259         ulimit.o                \
```

```
260         umask.o                 \
261         umount2.o               \
262         utssys.o                \
263         uucopy.o                \
264         vhangup.o               \
265         waitid.o                \
266         write.o                 \
267         writev.o                \
268         yield.o

270 SYSOBJS=                       \
271         __clock_gettime.o       \
272         __getcontext.o          \
273         __uadmin.o              \
274         _lwp_mutex_unlock.o     \
275         _stack_grow.o           \
276         door.o                  \
277         forkx.o                 \
278         forkallx.o              \
279         getcontext.o            \
280         gettimeofday.o          \
281         lwp_private.o           \
282         nuname.o                \
283         syscall.o               \
284         sysi86.o                \
285         tls_get_addr.o          \
286         uadmin.o                \
287         umount.o                \
288         uname.o                 \
289         vforkx.o

291 # Preserved solely to ease maintenance of 32-bit and 64-bit library builds
292 # This macro should ALWAYS be empty; native APIs are already 'large file'.
293 PORTGEN64=

295 # objects from source under $(LIBCDIR)/port
296 PORTFP=                         \
297         __flt_decim.o           \
298         __flt_rounds.o          \
299         __tbl_10_b.o            \
300         __tbl_10_h.o            \
301         __tbl_10_s.o            \
302         __tbl_2_b.o             \
303         __tbl_2_h.o             \
304         __tbl_2_s.o             \
305         __tbl_fdq.o             \
306         __tbl_tens.o            \
307         __x_power.o             \
308         _base_sup.o             \
309         aconvert.o              \
310         decimal_bin.o           \
311         double_decim.o          \
312         econvert.o              \
313         fconvert.o              \
314         file_decim.o            \
315         finite.o                \
316         fp_data.o               \
317         func_decim.o            \
318         gconvert.o              \
319         hex_bin.o               \
320         ieee_globals.o          \
321         pack_float.o            \
322         sigfpe.o                \
323         string_decim.o

325 PORTGEN=                        \
```



```
326         _env_data.o             \
327         _xftw.o                 \
328         a64l.o                  \
329         abort.o                 \
330         addsev.o                \
331         ascii_strcasecmp.o      \
332         ascii_strncasecmp.o     \
333         assert.o                \
334         atof.o                  \
335         atoi.o                  \
336         atol.o                  \
337         atoll.o                 \
338         attropen.o              \
339         atexit.o                \
340         atfork.o                \
341         basename.o              \
342         calloc.o                \
343         catgets.o               \
344         catopen.o               \
345         cfgetispeed.o           \
346         cfgetospeed.o           \
347         cfree.o                 \
348         cfsetispeed.o           \
349         cfsetospeed.o           \
350         cftime.o                \
351         clock.o                 \
352         closedir.o              \
353         closefrom.o             \
354         confstr.o               \
355         crypt.o                 \
356         csetlen.o               \
357         ctime.o                 \
358         ctime_r.o               \
359         daemon.o                \
360         deflt.o                 \
361         directio.o              \
362         dirname.o               \
363         div.o                   \
364         drand48.o               \
365         dup.o                   \
366         env_data.o              \
367         err.o                   \
368         errno.o                 \
369         euclen.o                \
370         event_port.o            \
371         execvp.o                \
372         fattach.o               \
373         fdetach.o               \
374         fdopendir.o             \
375         ffs.o                   \
376         fls.o                   \
377         fmtmsg.o                \
378         ftime.o                 \
379         ftok.o                  \
380         ftw.o                   \
381         gcvt.o                  \
382         getauxv.o               \
383         getcwd.o                \
384         getdate_err.o           \
385         getdtblsize.o           \
386         getenv.o                \
387         getexecname.o           \
388         getgrnam.o              \
389         getgrnam_r.o            \
390         gethostid.o             \
391         gethostname.o           \
```

```
392         gethz.o                 \
393         getisax.o               \
394         getloadavg.o            \
395         getlogin.o              \
396         getmntent.o             \
397         getnetgrent.o           \
398         get_nprocs.o            \
399         getopt.o                \
400         getopt_long.o           \
401         getpagesize.o           \
402         getpw.o                 \
403         getpwnam.o              \
404         getpwnam_r.o            \
405         getrusage.o             \
406         getspent.o              \
407         getspent_r.o            \
408         getsubopt.o             \
409         gettxt.o                \
410         getusershell.o          \
411         getut.o                 \
412         getutx.o                \
413         getvfsent.o             \
414         getwd.o                 \
415         getwidth.o              \
416         getxby_door.o           \
417         gtxt.o                  \
418         hsearch.o               \
419         iconv.o                 \
420         imaxabs.o               \
421         imaxdiv.o               \
422         index.o                 \
423         initgroups.o            \
424         insque.o                \
425         isaexec.o               \
426         isastream.o             \
427         isatty.o                \
428         killpg.o                \
429         klpdlib.o               \
430         l64a.o                  \
431         lckpwdf.o               \
432         lconstants.o            \
433         lexp10.o                \
434         lfind.o                 \
435         lfmt.o                  \
436         lfmt_log.o              \
437         lldiv.o                 \
438         llog10.o                \
439         lltostr.o               \
440         lmath.o                 \
441         localtime.o             \
442         lsearch.o               \
443         madvise.o               \
444         malloc.o                \
445         memalign.o              \
446         memmem.o                \
447         mkdev.o                 \
448         mkdtemp.o               \
449         mkfifo.o                \
450         mkstemp.o               \
451         mktemp.o                \
452         mlock.o                 \
453         mlockall.o              \
454         mon.o                   \
455         msync.o                 \
456         munlock.o               \
457         munlockall.o            \
```



```
458         ndbm.o                  \
459         nftw.o                  \
460         nlspath_checks.o        \
461         nsparse.o               \
462         nss_common.o            \
463         nss_dbdefs.o            \
464         nss_deffinder.o         \
465         opendir.o               \
466         opt_data.o              \
467         perror.o                \
468         pfmt.o                  \
469         pfmt_data.o             \
470         pfmt_print.o            \
471         pipe.o                  \
472         plock.o                 \
473         poll.o                  \
474         posix_fadvise.o         \
475         posix_fallocate.o       \
476         posix_madvise.o         \
477         posix_memalign.o        \
478         priocntl.o              \
479         privlib.o               \
480         priv_str_xlate.o        \
481         psiginfo.o              \
482         psignal.o               \
483         pt.o                    \
484         putpwent.o              \
485         putspent.o              \
486         raise.o                 \
487         rand.o                  \
488         random.o                \
489         rctlops.o               \
490         readdir.o               \
491         readdir_r.o             \
492         realpath.o              \
493         reboot.o                \
494         regexpr.o               \
495         remove.o                \
496         rewinddir.o             \
497         rindex.o                \
498         scandir.o               \
499         seekdir.o               \
500         select.o                \
501         setlabel.o              \
502         setpriority.o           \
503         settimeofday.o          \
504         sh_locks.o              \
505         sigflag.o               \
506         siglist.o               \
507         sigsend.o               \
508         sigsetops.o             \
509         ssignal.o               \
510         stack.o                 \
511         stpcpy.o                \
512         stpncpy.o               \
513         str2sig.o               \
514         strcase_charmap.o       \
515         strcat.o                \
516         strchr.o                \
517         strchrnul.o             \
518         strcspn.o               \
519         strdup.o                \
520         strerror.o              \
521         strlcat.o               \
522         strlcpy.o               \
523         strncat.o               \
```

```
524         strndup.o               \
525         strpbrk.o               \
526         strrchr.o               \
527         strsep.o                \
528         strsignal.o             \
529         strspn.o                \
530         strstr.o                \
531         strtod.o                \
532         strtoimax.o             \
533         strtok.o                \
534         strtok_r.o              \
535         strtoumax.o             \
536         swab.o                  \
537         swapctl.o               \
538         sysconf.o               \
539         syslog.o                \
540         tcdrain.o               \
541         tcflow.o                \
542         tcflush.o               \
543         tcgetattr.o             \
544         tcgetpgrp.o             \
545         tcgetsid.o              \
546         tcsendbreak.o           \
547         tcsetattr.o             \
548         tcsetpgrp.o             \
549         tell.o                  \
550         telldir.o               \
551         tfind.o                 \
552         time_data.o             \
553         time_gdata.o            \
554         tls_data.o              \
555         truncate.o              \
556         tsdalloc.o              \
557         tsearch.o               \
558         ttyname.o               \
559         ttyslot.o               \
560         ualarm.o                \
561         ucred.o                 \
562         valloc.o                \
563         vlfmt.o                 \
564         vpfmt.o                 \
565         waitpid.o               \
566         walkstack.o             \
567         wdata.o                 \
568         xgetwidth.o             \
569         xpg4.o                  \
570         xpg6.o

572 PORTPRINT_W=                   \
573         doprnt_w.o

575 PORTPRINT=                     \
576         asprintf.o              \
577         doprnt.o                \
578         fprintf.o               \
579         printf.o                \
580         snprintf.o              \
581         sprintf.o               \
582         vfprintf.o              \
583         vprintf.o               \
584         vsnprintf.o             \
585         vsprintf.o              \
586         vwprintf.o              \
587         wprintf.o

589 # Preserved solely to ease maintenance of 32-bit and 64-bit library builds
```



```
590 # This macro should ALWAYS be empty; native APIs are already 'large file'.
591 PORTSTDIO64=

593 PORTSTDIO_W=                   \
594         doscan_w.o

596 PORTSTDIO=                     \
597         __extensions.o          \
598         _endopen.o              \
599         _filbuf.o               \
600         _findbuf.o              \
601         _flsbuf.o               \
602         _wrtchk.o               \
603         clearerr.o              \
604         ctermid.o               \
605         ctermid_r.o             \
606         cuserid.o               \
607         data.o                  \
608         doscan.o                \
609         fdopen.o                \
610         feof.o                  \
611         ferror.o                \
612         fgetc.o                 \
613         fgets.o                 \
614         fileno.o                \
615         flockf.o                \
616         flush.o                 \
617         fopen.o                 \
618         fpos.o                  \
619         fputc.o                 \
620         fputs.o                 \
621         fread.o                 \
622         fseek.o                 \
623         fseeko.o                \
624         ftell.o                 \
625         ftello.o                \
626         fwrite.o                \
627         getc.o                  \
628         getchar.o               \
629         getline.o               \
630         getpass.o               \
631         gets.o                  \
632         getw.o                  \
633         mse.o                   \
634         popen.o                 \
635         putc.o                  \
636         putchar.o               \
637         puts.o                  \
638         putw.o                  \
639         rewind.o                \
640         scanf.o                 \
641         setbuf.o                \
642         setbuffer.o             \
643         setvbuf.o               \
644         system.o                \
645         tempnam.o               \
646         tmpfile.o               \
647         tmpnam_r.o              \
648         ungetc.o                \
649         vscanf.o                \
650         vwscanf.o               \
651         wscanf.o

653 PORTI18N=                      \
654         getwchar.o              \
655         putwchar.o              \
```

```
656         putws.o                 \
657         strcasecmp.o            \
658         strcasestr.o            \
659         strncasecmp.o           \
660         strtows.o               \
661         wcsnlen.o               \
662         wcsstr.o                \
663         wcstoimax.o             \
664         wcstol.o                \
665         wcstoul.o               \
666         wcswcs.o                \
667         wmemchr.o               \
668         wmemcmp.o               \
669         wmemcpy.o               \
670         wmemmove.o              \
671         wmemset.o               \
672         wscasecmp.o             \
673         wscat.o                 \
674         wschr.o                 \
675         wscmp.o                 \
676         wscpy.o                 \
677         wscspn.o                \
678         wsdup.o                 \
679         wslen.o                 \
680         wsncasecmp.o            \
681         wsncat.o                \
682         wsncmp.o                \
683         wsncpy.o                \
684         wspbrk.o                \
685         wsprintf.o              \
686         wsrchr.o                \
687         wsscanf.o               \
688         wsspn.o                 \
689         wstod.o                 \
690         wstok.o                 \
691         wstol.o                 \
692         wstoll.o                \
693         wsxfrm.o                \
694         gettext.o               \
695         gettext_gnu.o           \
696         gettext_real.o          \
697         gettext_util.o          \
698         isdigit.o               \
699         plural_parser.o         \
700         wdresolve.o             \
701         _ctype.o                \
702         isascii.o               \
703         toascii.o

705 PORTI18N_COND=                  \
706         wcstol_longlong.o       \
707         wcstoul_longlong.o

709 PORTLOCALE=                    \
710         big5.o                  \
711         btowc.o                 \
712         collate.o               \
713         collcmp.o               \
714         euc.o                   \
715         fnmatch.o               \
716         fgetwc.o                \
717         fgetws.o                \
718         fix_grouping.o          \
719         fputwc.o                \
720         fputws.o                \
721         fwide.o                 \
```



```
722         gb18030.o               \
723         gb2312.o                \
724         gbk.o                   \
725         getdate.o               \
726         iswctype.o              \
727         ldpart.o                \
728         lmessages.o             \
729         lnumeric.o              \
730         lmonetary.o             \
731         localeconv.o            \
732         mbftowc.o               \
733         mblen.o                 \
734         mbrlen.o                \
735         mbrtowc.o               \
736         mbsinit.o               \
737         mbsnrtowcs.o            \
738         mbsrtowcs.o             \
739         mbstowcs.o              \
740         mbtowc.o                \
741         mskanji.o               \
742         nextwctype.o            \
743         nl_langinfo.o           \
744         none.o                  \
745         regcomp.o               \
746         regfree.o               \
747         regerror.o              \
748         regexec.o               \
749         rune.o                  \
750         runetype.o              \
751         setlocale.o             \
752         setrunelocale.o         \
753         strcoll.o               \
754         strfmon.o               \
755         strftime.o              \
756         strptime.o              \
757         strxfrm.o               \
758         table.o                 \
759         timelocal.o             \
760         tolower.o               \
761         towlower.o              \
762         ungetwc.o               \
763         utf8.o                  \
764         wcrtomb.o               \
765         wcscoll.o               \
766         wcsftime.o              \
767         wcsnrtombs.o            \
768         wcsrtombs.o             \
769         wcswidth.o              \
770         wcstombs.o              \
771         wcsxfrm.o               \
772         wctob.o                 \
773         wctomb.o                \
774         wctrans.o               \
775         wctype.o                \
776         wcwidth.o               \
777         wscol.o                 \
778         xlocale.o
777         wscol.o

780 AIOOBJS=                       \
781         aio.o                   \
782         aio_alloc.o             \
783         posix_aio.o

785 RTOBJS=                        \
786         clock_timer.o           \
```

```
787         mqueue.o                \
788         pos4obj.o               \
789         sched.o                 \
790         sem.o                   \
791         shm.o                   \
792         sigev_thread.o

794 TPOOLOBJS=                       \
795         thread_pool.o

797 THREADSOBJS=                     \
798         alloc.o                 \
799         assfail.o               \
800         cancel.o                \
801         door_calls.o            \
802         pthr_attr.o             \
803         pthr_barrier.o          \
804         pthr_cond.o             \
805         pthr_mutex.o            \
806         pthr_rwlock.o           \
807         pthread.o               \
808         rwlock.o                \
809         scalls.o                \
810         sema.o                  \
811         sigaction.o             \
812         spawn.o                 \
813         synch.o                 \
814         tdb_agent.o             \
815         thr.o                   \
816         thread_interface.o      \
817         tls.o                   \
818         tsd.o

820 THREADSMACHOBJS=                \
821         machdep.o

823 THREADSASMOBJS=                 \
824         asm_subr.o

826 UNICODEOBJS=                    \
827         u8_textprep.o           \
828         uconv.o

830 UNWINDMACHOBJS=                 \
831         call_frame_inst.o       \
832         eh_frame.o              \
833         thrp_unwind.o           \
834         unwind.o

836 pics/unwind.o:= COPTFLAG64 =

838 UNWINDASMOBJS=                  \
839         unwind_frame.o

841 # Preserved solely to ease maintenance of 32-bit and 64-bit library builds
842 # This macro should ALWAYS be empty; native APIs are already 'large file'.
843 PORTSYS64=

845 PORTSYS=                        \
846         _autofssys.o            \
847         access.o                \
848         acctctl.o               \
849         bsd_signal.o            \
850         chmod.o                 \
851         chown.o                 \
852         corectl.o               \
```



```
853         exacctsys.o             \
854         execl.o                 \
855         execle.o                \
856         execv.o                 \
857         fcntl.o                 \
858         getpagesizes.o          \
859         getpeerucred.o          \
860         inst_sync.o             \
861         issetugid.o             \
862         label.o                 \
863         link.o                  \
864         lockf.o                 \
865         lwp.o                   \
866         lwp_cond.o              \
867         lwp_rwlock.o            \
868         lwp_sigmask.o           \
869         meminfosys.o            \
870         mkdir.o                 \
871         mknod.o                 \
872         msgsys.o                \
873         nfssys.o                \
874         open.o                  \
875         pgrpsys.o               \
876         posix_sigwait.o         \
877         ppriv.o                 \
878         psetsys.o               \
879         rctlsys.o               \
880         readlink.o              \
881         rename.o                \
882         sbrk.o                  \
883         semsys.o                \
884         set_errno.o             \
885         sharefs.o               \
886         shmsys.o                \
887         sidsys.o                \
888         siginterrupt.o          \
889         signal.o                \
890         sigpending.o            \
891         sigstack.o              \
892         stat.o                  \
893         symlink.o               \
894         tasksys.o               \
895         time.o                  \
896         time_util.o             \
897         ucontext.o              \
898         unlink.o                \
899         ustat.o                 \
900         utimesys.o              \
901         zone.o

903 PORTREGEX=                      \
904         glob.o                  \
905         regcmp.o                \
906         regex.o                 \
907         wordexp.o

909 VALUES=                        \
910         values-Xa.o

912 MOSTOBJS=                      \
913         $(STRETS)               \
914         $(CRTOBJS)              \
915         $(DYNOBJS)              \
916         $(FPOBJS)               \
917         $(I386FPOBJS)           \
918         $(FPASMOBJS)            \
```

```
 919         $(ATOMICOBJS)           \
 920         $(XATTROBJS)            \
 921         $(COMOBJS)              \
 922         $(GENOBJS)              \
 923         $(PORTFP)               \
 924         $(PORTGEN)              \
 925         $(PORTGEN64)            \
 926         $(PORTI18N)             \
 927         $(PORTI18N_COND)        \
 928         $(PORTLOCALE)           \
 929         $(PORTPRINT)            \
 930         $(PORTPRINT_W)          \
 931         $(PORTREGEX)            \
 932         $(PORTSTDIO)            \
 933         $(PORTSTDIO64)          \
 934         $(PORTSTDIO_W)          \
 935         $(PORTSYS)              \
 936         $(PORTSYS64)            \
 937         $(AIOOBJS)              \
 938         $(RTOBJS)               \
 939         $(TPOOLOBJS)            \
 940         $(THREADSOBJS)          \
 941         $(THREADSMACHOBJS)      \
 942         $(THREADSASMOBJS)       \
 943         $(UNICODEOBJS)          \
 944         $(UNWINDMACHOBJS)       \
 945         $(UNWINDASMOBJS)        \
 946         $(COMSYSOBJS)           \
 947         $(SYSOBJS)              \
 948         $(COMSYSOBJS64)         \
 949         $(SYSOBJS64)            \
 950         $(VALUES)

 952 TRACEOBJS=                      \
 953         plockstat.o

 955 # NOTE: libc.so.1 must be linked with the minimal crti.o and crtn.o
 956 # modules whose source is provided in the $(SRC)/lib/common directory.
 957 # This must be done because otherwise the Sun C compiler would insert
 958 # its own versions of these modules and those versions contain code
 959 # to call out to C++ initialization functions.  Such C++ initialization
 960 # functions can call back into libc before thread initialization is
 961 # complete and this leads to segmentation violations and other problems.
 962 # Since libc contains no C++ code, linking with the minimal crti.o and
 963 # crtn.o modules is safe and avoids the problems described above.
 964 OBJECTS= $(CRTI) $(MOSTOBJS) $(CRTN)
 965 CRTSRCS= ../../common/amd64

 967 # include common library definitions
 968 include ../../Makefile.lib
 969 include ../../Makefile.lib.64

 971 CFLAGS64 += $(CTF_FLAGS)

 973 # This is necessary to avoid problems with calling _ex_unwind().
 974 # We probably don't want any inlining anyway.
 975 CFLAGS64 += -xinline=

 977 CERRWARN += -_gcc=-Wno-parentheses
 978 CERRWARN += -_gcc=-Wno-switch
 979 CERRWARN += -_gcc=-Wno-uninitialized
 980 CERRWARN += -_gcc=-Wno-unused-value
 981 CERRWARN += -_gcc=-Wno-unused-label
 982 CERRWARN += -_gcc=-Wno-unused-variable
 983 CERRWARN += -_gcc=-Wno-type-limits
 984 CERRWARN += -_gcc=-Wno-char-subscripts
```



```
 985 CERRWARN += -_gcc=-Wno-clobbered
 986 CERRWARN += -_gcc=-Wno-unused-function
 987 CERRWARN += -_gcc=-Wno-address

 989 # Setting THREAD_DEBUG = -DTHREAD_DEBUG (make THREAD_DEBUG=-DTHREAD_DEBUG ...)
 990 # enables ASSERT() checking in the threads portion of the library.
 991 # This is automatically enabled for DEBUG builds, not for non-debug builds.
 992 THREAD_DEBUG =
 993 $(NOT_RELEASE_BUILD)THREAD_DEBUG = -DTHREAD_DEBUG

 995 # Make string literals read-only to save memory
 996 CFLAGS64 += $(XSTRCONST)

 998 ALTPICS= $(TRACEOBJS:%=pics/%)

1000 $(DYNLIB) := BUILD.SO = $(LD) -o $@ -G $(DYNFLAGS) $(PICS) $(ALTPICS) $(EXTPICS)

1002 MAPFILES =      $(LIBCDIR)/port/mapfile-vers

1004 CPPFLAGS=       -D_REENTRANT -D$(MACH64) -D__$(MACH64) $(THREAD_DEBUG) \
1005                 -I. -I$(LIBCBASE)/inc -I$(LIBCDIR)/inc $(CPPFLAGS.master)
1006 ASFLAGS=        $(AS_PICFLAGS) -P -D__STDC__ -D_ASM $(CPPFLAGS) \
1007                 $(amd64_AS_XARCH)

1009 # As a favor to the dtrace syscall provider, libc still calls the
1010 # old syscall traps that have been obsoleted by the *at() interfaces.
1011 # Delete this to compile libc using only the new *at() system call traps
1012 CPPFLAGS += -D_RETAIN_OLD_SYSCALLS

1014 # proc64_id.o is built with defines in $(SRC)/uts/intel/sys/x86_archext.h
1015 pics/proc64_id.o        := CFLAGS64 += -I$(SRC)/uts/intel

1017 # Inform the run-time linker about libc specialized initialization
1018 RTLDINFO =      -z rtldinfo=tls_rtldinfo
1019 DYNFLAGS +=     $(RTLDINFO)

1021 # Force libc's internal references to be resolved immediately upon loading
1022 # in order to avoid critical region problems.  Since almost all libc symbols
1023 # are marked 'protected' in the mapfiles, this is a minimal set (15 to 20).
1024 DYNFLAGS +=     -znow

1026 BUILD.s=        $(AS) $(ASFLAGS) $< -o $@

1028 # Override this top level flag so the compiler builds in its native
1029 # C99 mode.  This has been enabled to support the complex arithmetic
1030 # added to libc.
1031 C99MODE=        $(C99_ENABLE)

1033 # libc method of building an archive
1034 # The "$(GREP) -v ' L '" part is necessary only until
1035 # lorder is fixed to ignore thread-local variables.
1036 BUILD.AR= $(RM) $@ ; \
1037         $(AR) q $@ `$(LORDER) $(MOSTOBJS:%=$(DIR)/%) | $(GREP) -v ' L ' | $(TSOR

1039 # extra files for the clean target
1040 CLEANFILES=                     \
1041         $(LIBCDIR)/port/gen/errlst.c    \
1042         $(LIBCDIR)/port/gen/new_list.c  \
1043         assym.h                 \
1044         genassym                \
1045         crt/_rtld.s             \
1046         pics/crti.o             \
1047         pics/crtn.o             \
1048         $(ALTPICS)

1050 CLOBBERFILES += $(LIB_PIC)
```

```
1052 # list of C source for lint
1053 SRCS=                                                  \
1054         $(ATOMICOBJS:%.o=$(SRC)/common/atomic/%.c)      \
1055         $(XATTROBJS:%.o=$(SRC)/common/xattr/%.c)        \
1056         $(COMOBJS:%.o=$(SRC)/common/util/%.c)           \
1057         $(PORTFP:%.o=$(LIBCDIR)/port/fp/%.c)                    \
1058         $(PORTGEN:%.o=$(LIBCDIR)/port/gen/%.c)                  \
1059         $(PORTI18N:%.o=$(LIBCDIR)/port/i18n/%.c)                \
1060         $(PORTLOCALE:%.o=$(LIBCDIR)/port/locale/%.c)            \
1061         $(PORTPRINT:%.o=$(LIBCDIR)/port/print/%.c)              \
1062         $(PORTREGEX:%.o=$(LIBCDIR)/port/regex/%.c)              \
1063         $(PORTSTDIO:%.o=$(LIBCDIR)/port/stdio/%.c)              \
1064         $(PORTSYS:%.o=$(LIBCDIR)/port/sys/%.c)                  \
1065         $(AIOOBJS:%.o=$(LIBCDIR)/port/aio/%.c)                  \
1066         $(RTOBJS:%.o=$(LIBCDIR)/port/rt/%.c)                    \
1067         $(TPOOLOBJS:%.o=$(LIBCDIR)/port/tpool/%.c)              \
1068         $(THREADSOBJS:%.o=$(LIBCDIR)/port/threads/%.c)          \
1069         $(THREADSMACHOBJS:%.o=threads/%.c)              \
1070         $(UNICODEOBJS:%.o=$(SRC)/common/unicode/%.c)    \
1071         $(UNWINDMACHOBJS:%.o=unwind/%.c)                \
1072         $(FPOBJS:%.o=fp/%.c)                            \
1073         $(I386FPOBJS:%.o=$(LIBCDIR)/i386/fp/%.c)                \
1074         $(LIBCBASE)/gen/ecvt.c                          \
1075         $(LIBCBASE)/gen/makectxt.c                      \
1076         $(LIBCBASE)/gen/siginfolst.c                    \
1077         $(LIBCBASE)/gen/siglongjmp.c                    \
1078         $(LIBCBASE)/gen/sync_instruction_memory.c       \
1079         $(LIBCBASE)/sys/uadmin.c

1081 # conditional assignments
1082 # $(DYNLIB) $(LIB_PIC) := DYNOBJS = _rtbootld.o
1083 $(DYNLIB) := CRTI = crti.o
1084 $(DYNLIB) := CRTN = crtn.o

1086 # Files which need the threads .il inline template
1087 TIL=                            \
1088         aio.o                   \
1089         alloc.o                 \
1090         assfail.o               \
1091         atexit.o                \
1092         atfork.o                \
1093         cancel.o                \
1094         door_calls.o            \
1095         err.o                   \
1096         errno.o                 \
1097         lwp.o                   \
1098         ma.o                    \
1099         machdep.o               \
1100         posix_aio.o             \
1101         pthr_attr.o             \
1102         pthr_barrier.o          \
1103         pthr_cond.o             \
1104         pthr_mutex.o            \
1105         pthr_rwlock.o           \
1106         pthread.o               \
1107         rand.o                  \
1108         rwlock.o                \
1109         scalls.o                \
1110         sched.o                 \
1111         sema.o                  \
1112         sigaction.o             \
1113         sigev_thread.o          \
1114         spawn.o                 \
1115         stack.o                 \
1116         synch.o                 \
```



```
1117         tdb_agent.o             \
1118         thr.o                   \
1119         thread_interface.o      \
1120         thread_pool.o           \
1121         thrp_unwind.o           \
1122         tls.o                   \
1123         tsd.o

1125 $(TIL:%=pics/%) := CFLAGS64 += $(LIBCBASE)/threads/amd64.il

1127 # pics/mul64.o := CFLAGS64 += crt/mul64.il

1129 # large-file-aware components that should be built large

1131 #$(COMSYSOBJS64:%=pics/%) := \
1132 #       CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1134 #$(SYSOBJS64:%=pics/%) := \
1135 #       CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1137 #$(PORTGEN64:%=pics/%) := \
1138 #       CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1140 #$(PORTSTDIO64:%=pics/%) := \
1141 #       CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1143 #$(PORTSYS64:%=pics/%) := \
1144 #       CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1146 $(PORTSTDIO_W:%=pics/%) := \
1147         CPPFLAGS += -D_WIDE

1149 $(PORTPRINT_W:%=pics/%) := \
1150         CPPFLAGS += -D_WIDE

1152 $(PORTPRINT_C89:%=pics/%) := \
1153         CPPFLAGS += -D_C89_INTMAX32

1155 $(PORTSTDIO_C89:%=pics/%) := \
1156         CPPFLAGS += -D_C89_INTMAX32

1158 $(PORTI18N_COND:%=pics/%) := \
1159         CPPFLAGS += -D_WCS_LONGLONG

1161 .KEEP_STATE:

1163 all: $(LIBS) $(LIB_PIC)

1165 lint    :=      CPPFLAGS += -I$(LIBCDIR)/$(MACH)/fp
1166 lint    :=      CPPFLAGS += -D_MSE_INT_H -D_LCONV_C99
1167 lint    :=      LINTFLAGS64 += -mn -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED

1169 lint:
1170         @echo $(LINT.c) ... $(LDLIBS)
1171         @$(LINT.c) $(SRCS) $(LDLIBS)

1173 $(LINTLIB):= SRCS=$(LIBCDIR)/port/llib-lc
1174 $(LINTLIB):= CPPFLAGS += -D_MSE_INT_H
1175 $(LINTLIB):= LINTFLAGS64=-nvx -m64

1177 # object files that depend on inline template
1178 $(TIL:%=pics/%): $(LIBCBASE)/threads/amd64.il
1179 # pics/mul64.o: crt/mul64.il

1181 # include common libc targets
1182 include ../Makefile.targ
```

```
1184 # We need to strip out all CTF data from the static library
1185 $(LIB_PIC) := DIR = pics
1186 $(LIB_PIC): pics $$(PICS)
1187         $(BUILD.AR)
1188         $(MCS) -d -n .SUNW_ctf $@ > /dev/null 2>&1
1189         $(AR) -ts $@ > /dev/null
1190         $(POST_PROCESS_A)

1192 ASSYMDEP_OBJS=                  \
1193         _lwp_mutex_unlock.o     \
1194         _stack_grow.o           \
1195         asm_subr.o              \
1196         getcontext.o            \
1197         setjmp.o                \
1198         tls_get_addr.o          \
1199         vforkx.o

1201 $(ASSYMDEP_OBJS:%=pics/%): assym.h

1203 # assym.h build rules

1205 GENASSYM_C = genassym.c

1207 genassym: $(GENASSYM_C)
1208         $(NATIVECC) -Iinc -I$(LIBCDIR)/inc $(CPPFLAGS.native)   \
1209                 -o $@ $(GENASSYM_C)

1211 OFFSETS = $(LIBCDIR)/$(MACH)/offsets.in

1213 assym.h: $(OFFSETS) genassym
1214         $(OFFSETS_CREATE) <$(OFFSETS) >$@
1215         ./genassym >>$@

1217 # derived C source and related explicit dependencies
1218 $(LIBCDIR)/port/gen/errlst.c + \
1219 $(LIBCDIR)/port/gen/new_list.c: $(LIBCDIR)/port/gen/errlist $(LIBCDIR)/port/gen/
1220         cd $(LIBCDIR)/port/gen; pwd; $(AWK) -f errlist.awk < errlist

1222 pics/errlst.o: $(LIBCDIR)/port/gen/errlst.c

1224 pics/new_list.o: $(LIBCDIR)/port/gen/new_list.c
```

```
**********************************************************
   23909 Thu Jul 11 12:26:10 2013
new/usr/src/lib/libc/i386/Makefile.com
2964 need POSIX 2008 locale object support
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 2004, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2013, OmniTI Computer Consulting, Inc. All rights reserved.
#
# Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
# Use is subject to license terms.
#

LIBCDIR=        $(SRC)/lib/libc
LIB_PIC=        libc_pic.a
VERS=           .1
CPP=            /usr/lib/cpp
TARGET_ARCH=    i386

VALUES=         values-Xa.o

# objects are grouped by source directory

# local objects
STRETS=

CRTOBJS=                        \
        cerror.o                \
        cerror64.o

DYNOBJS=                        \
        _rtbootld.o

FPOBJS=                         \
        _D_cplx_div.o           \
        _D_cplx_div_ix.o        \
        _D_cplx_div_rx.o        \
        _D_cplx_lr_div.o        \
        _D_cplx_lr_div_ix.o     \
        _D_cplx_lr_div_rx.o     \
        _D_cplx_mul.o           \
        _F_cplx_div.o           \
        _F_cplx_div_ix.o        \
        _F_cplx_div_rx.o        \
        _F_cplx_lr_div.o        \
        _F_cplx_lr_div_ix.o     \
```



```
        _F_cplx_lr_div_rx.o     \
        _F_cplx_mul.o           \
        _X_cplx_div.o           \
        _X_cplx_div_ix.o        \
        _X_cplx_div_rx.o        \
        _X_cplx_lr_div.o        \
        _X_cplx_lr_div_ix.o     \
        _X_cplx_lr_div_rx.o     \
        _X_cplx_mul.o           \
        _base_il.o              \
        fpgetmask.o             \
        fpgetround.o            \
        fpgetsticky.o           \
        fpsetmask.o             \
        fpsetround.o            \
        fpsetsticky.o           \
        fpstart.o

FPASMOBJS=                      \
        __xgetRD.o              \
        _xtoll.o                \
        _xtoull.o               \
        fpcw.o

ATOMICOBJS=                     \
        atomic.o

XATTROBJS=                      \
        xattr_common.o

COMOBJS=                        \
        bcmp.o                  \
        bcopy.o                 \
        bsearch.o               \
        bzero.o                 \
        qsort.o                 \
        strtol.o                \
        strtoul.o               \
        strtoll.o               \
        strtoull.o

DTRACEOBJS=                     \
        dtrace_data.o

GENOBJS=                        \
        _div64.o                \
        _divdi3.o               \
        _getsp.o                \
        _mul64.o                \
        abs.o                   \
        alloca.o                \
        byteorder.o             \
        byteorder64.o           \
        cuexit.o                \
        ecvt.o                  \
        errlst.o                \
        i386_data.o             \
        ladd.o                  \
        ldivide.o               \
        lmul.o                  \
        lock.o                  \
        lshiftl.o               \
        lsign.o                 \
        lsub.o                  \
        makectxt.o              \
        memccpy.o               \
```

```
128         memchr.o                \
129         memcmp.o                \
130         memcpy.o                \
131         memset.o                \
132         new_list.o              \
133         setjmp.o                \
134         siginfolst.o            \
135         siglongjmp.o            \
136         strcat.o                \
137         strchr.o                \
138         strcmp.o                \
139         strcpy.o                \
140         strlen.o                \
141         strncat.o               \
142         strncmp.o               \
143         strncpy.o               \
144         strnlen.o               \
145         strrchr.o               \
146         sync_instruction_memory.o

148 # sysobjs that contain large-file interfaces
149 COMSYSOBJS64=                   \
150         fstatvfs64.o            \
151         getdents64.o            \
152         getrlimit64.o           \
153         lseek64.o               \
154         mmap64.o                \
155         pread64.o               \
156         pwrite64.o              \
157         setrlimit64.o           \
158         statvfs64.o

160 SYSOBJS64=

162 COMSYSOBJS=                     \
163         __clock_timer.o         \
164         __getloadavg.o          \
165         __rusagesys.o           \
166         __signotify.o           \
167         __sigrt.o               \
168         __time.o                \
169         _lgrp_home_fast.o       \
170         _lgrpsys.o              \
171         _nfssys.o               \
172         _portfs.o               \
173         _pset.o                 \
174         _rpcsys.o               \
175         _sigaction.o            \
176         _so_accept.o            \
177         _so_bind.o              \
178         _so_connect.o           \
179         _so_getpeername.o       \
180         _so_getsockname.o       \
181         _so_getsockopt.o        \
182         _so_listen.o            \
183         _so_recv.o              \
184         _so_recvfrom.o          \
185         _so_recvmsg.o           \
186         _so_send.o              \
187         _so_sendmsg.o           \
188         _so_sendto.o            \
189         _so_setsockopt.o        \
190         _so_shutdown.o          \
191         _so_socket.o            \
192         _so_socketpair.o        \
193         _sockconfig.o           \
```



```
194         acct.o                  \
195         acl.o                   \
196         adjtime.o               \
197         alarm.o                 \
198         brk.o                   \
199         chdir.o                 \
200         chroot.o                \
201         cladm.o                 \
202         close.o                 \
203         execve.o                \
204         exit.o                  \
205         facl.o                  \
206         fchdir.o                \
207         fchroot.o               \
208         fdsync.o                \
209         fpathconf.o             \
210         fstatfs.o               \
211         fstatvfs.o              \
212         getcpuid.o              \
213         getdents.o              \
214         getegid.o               \
215         geteuid.o               \
216         getgid.o                \
217         getgroups.o             \
218         gethrtime.o             \
219         getitimer.o             \
220         getmsg.o                \
221         getpid.o                \
222         getpmsg.o               \
223         getppid.o               \
224         getrlimit.o             \
225         getuid.o                \
226         gtty.o                  \
227         install_utrap.o         \
228         ioctl.o                 \
229         kaio.o                  \
230         kill.o                  \
231         llseek.o                \
232         lseek.o                 \
233         mmapobjsys.o            \
234         memcntl.o               \
235         mincore.o               \
236         mmap.o                  \
237         modctl.o                \
238         mount.o                 \
239         mprotect.o              \
240         munmap.o                \
241         nice.o                  \
242         ntp_adjtime.o           \
243         ntp_gettime.o           \
244         p_online.o              \
245         pathconf.o              \
246         pause.o                 \
247         pcsample.o              \
248         pipe2.o                 \
249         pollsys.o               \
250         pread.o                 \
251         priocntlset.o           \
252         processor_bind.o        \
253         processor_info.o        \
254         profil.o                \
255         putmsg.o                \
256         putpmsg.o               \
257         pwrite.o                \
258         read.o                  \
259         readv.o                 \
```

```
260         resolvepath.o           \
261         seteguid.o              \
262         setgid.o                \
263         setgroups.o             \
264         setitimer.o             \
265         setreid.o               \
266         setrlimit.o             \
267         setuid.o                \
268         sigaltstk.o             \
269         sigprocmsk.o            \
270         sigsendset.o            \
271         sigsuspend.o            \
272         statfs.o                \
273         statvfs.o               \
274         stty.o                  \
275         sync.o                  \
276         sysconfig.o             \
277         sysfs.o                 \
278         sysinfo.o               \
279         syslwp.o                \
280         times.o                 \
281         ulimit.o                \
282         umask.o                 \
283         umount2.o               \
284         utssys.o                \
285         uucopy.o                \
286         vhangup.o               \
287         waitid.o                \
288         write.o                 \
289         writev.o                \
290         yield.o

292 SYSOBJS=                        \
293         __clock_gettime.o       \
294         __getcontext.o          \
295         __uadmin.o              \
296         _lwp_mutex_unlock.o     \
297         _stack_grow.o           \
298         door.o                  \
299         forkx.o                 \
300         forkallx.o              \
301         getcontext.o            \
302         gettimeofday.o          \
303         lwp_private.o           \
304         nuname.o                \
305         ptrace.o                \
306         syscall.o               \
307         sysi86.o                \
308         tls_get_addr.o          \
309         uadmin.o                \
310         umount.o                \
311         uname.o                 \
312         vforkx.o                \
313         xstat.o

315 # objects under $(LIBCDIR)/port which contain transitional large file interfaces
316 PORTGEN64=                      \
317         _xftw64.o               \
318         attropen64.o            \
319         ftw64.o                 \
320         mkstemp64.o             \
321         nftw64.o                \
322         tell64.o                \
323         truncate64.o

325 # objects from source under $(LIBCDIR)/port
```



```
326 PORTFP=                         \
327         __flt_decim.o           \
328         __flt_rounds.o          \
329         __tbl_10_b.o            \
330         __tbl_10_h.o            \
331         __tbl_10_s.o            \
332         __tbl_2_b.o             \
333         __tbl_2_h.o             \
334         __tbl_2_s.o             \
335         __tbl_fdq.o             \
336         __tbl_tens.o            \
337         __x_power.o             \
338         _base_sup.o             \
339         aconvert.o              \
340         decimal_bin.o           \
341         double_decim.o          \
342         econvert.o              \
343         fconvert.o              \
344         file_decim.o            \
345         finite.o                \
346         fp_data.o               \
347         func_decim.o            \
348         gconvert.o              \
349         hex_bin.o               \
350         ieee_globals.o          \
351         pack_float.o            \
352         sigfpe.o                \
353         string_decim.o

355 PORTGEN=                        \
356         _env_data.o             \
357         _xftw.o                 \
358         a64l.o                  \
359         abort.o                 \
360         addsev.o                \
361         ascii_strcasecmp.o      \
362         ascii_strncasecmp.o     \
363         assert.o                \
364         atof.o                  \
365         atoi.o                  \
366         atol.o                  \
367         atoll.o                 \
368         attrat.o                \
369         attropen.o              \
370         atexit.o                \
371         atfork.o                \
372         basename.o              \
373         calloc.o                \
374         catgets.o               \
375         catopen.o               \
376         cfgetispeed.o           \
377         cfgetospeed.o           \
378         cfree.o                 \
379         cfsetispeed.o           \
380         cfsetospeed.o           \
381         cftime.o                \
382         clock.o                 \
383         closedir.o              \
384         closefrom.o             \
385         confstr.o               \
386         crypt.o                 \
387         csetlen.o               \
388         ctime.o                 \
389         ctime_r.o               \
390         daemon.o                \
391         deflt.o                 \
```

```
392         directio.o              \
393         dirname.o               \
394         div.o                   \
395         drand48.o               \
396         dup.o                   \
397         env_data.o              \
398         err.o                   \
399         errno.o                 \
400         euclen.o                \
401         event_port.o            \
402         execvp.o                \
403         fattach.o               \
404         fdetach.o               \
405         fdopendir.o             \
406         ffs.o                   \
407         fls.o                   \
408         fmtmsg.o                \
409         ftime.o                 \
410         ftok.o                  \
411         ftw.o                   \
412         gcvt.o                  \
413         getauxv.o               \
414         getcwd.o                \
415         getdate_err.o           \
416         getdtblsize.o           \
417         getenv.o                \
418         getexecname.o           \
419         getgrnam.o              \
420         getgrnam_r.o            \
421         gethostid.o             \
422         gethostname.o           \
423         gethz.o                 \
424         getisax.o               \
425         getloadavg.o            \
426         getlogin.o              \
427         getmntent.o             \
428         getnetgrent.o           \
429         get_nprocs.o            \
430         getopt.o                \
431         getopt_long.o           \
432         getpagesize.o           \
433         getpw.o                 \
434         getpwnam.o              \
435         getpwnam_r.o            \
436         getrusage.o             \
437         getspent.o              \
438         getspent_r.o            \
439         getsubopt.o             \
440         gettxt.o                \
441         getusershell.o          \
442         getut.o                 \
443         getutx.o                \
444         getvfsent.o             \
445         getwd.o                 \
446         getwidth.o              \
447         getxby_door.o           \
448         gtxt.o                  \
449         hsearch.o               \
450         iconv.o                 \
451         imaxabs.o               \
452         imaxdiv.o               \
453         index.o                 \
454         initgroups.o            \
455         insque.o                \
456         isaexec.o               \
457         isastream.o             \
```



```
458         isatty.o                \
459         killpg.o                \
460         klpdlib.o               \
461         l64a.o                  \
462         lckpwdf.o               \
463         lconstants.o            \
464         lexp10.o                \
465         lfind.o                 \
466         lfmt.o                  \
467         lfmt_log.o              \
468         llabs.o                 \
469         lldiv.o                 \
470         llog10.o                \
471         lltostr.o               \
472         localtime.o             \
473         lsearch.o               \
474         madvise.o               \
475         malloc.o                \
476         memalign.o              \
477         memmem.o                \
478         mkdev.o                 \
479         mkdtemp.o               \
480         mkfifo.o                \
481         mkstemp.o               \
482         mktemp.o                \
483         mlock.o                 \
484         mlockall.o              \
485         mon.o                   \
486         msync.o                 \
487         munlock.o               \
488         munlockall.o            \
489         ndbm.o                  \
490         nftw.o                  \
491         nlspath_checks.o        \
492         nsparse.o               \
493         nss_common.o            \
494         nss_dbdefs.o            \
495         nss_deffinder.o         \
496         opendir.o               \
497         opt_data.o              \
498         perror.o                \
499         pfmt.o                  \
500         pfmt_data.o             \
501         pfmt_print.o            \
502         pipe.o                  \
503         plock.o                 \
504         poll.o                  \
505         posix_fadvise.o         \
506         posix_fallocate.o       \
507         posix_madvise.o         \
508         posix_memalign.o        \
509         priocntl.o              \
510         privlib.o               \
511         priv_str_xlate.o        \
512         psiginfo.o              \
513         psignal.o               \
514         pt.o                    \
515         putpwent.o              \
516         putspent.o              \
517         raise.o                 \
518         rand.o                  \
519         random.o                \
520         rctlops.o               \
521         readdir.o               \
522         readdir_r.o             \
523         realpath.o              \
```

```
524         reboot.o                \
525         regexpr.o               \
526         remove.o                \
527         rewinddir.o             \
528         rindex.o                \
529         scandir.o               \
530         seekdir.o               \
531         select.o                \
532         select_large_fdset.o    \
533         setlabel.o              \
534         setpriority.o           \
535         settimeofday.o          \
536         sh_locks.o              \
537         sigflag.o               \
538         siglist.o               \
539         sigsend.o               \
540         sigsetops.o             \
541         ssignal.o               \
542         stack.o                 \
543         stpcpy.o                \
544         stpncpy.o               \
545         str2sig.o               \
546         strcase_charmap.o       \
547         strchrnul.o             \
548         strcspn.o               \
549         strdup.o                \
550         strerror.o              \
551         strlcat.o               \
552         strlcpy.o               \
553         strndup.o               \
554         strpbrk.o               \
555         strsep.o                \
556         strsignal.o             \
557         strspn.o                \
558         strstr.o                \
559         strtod.o                \
560         strtoimax.o             \
561         strtok.o                \
562         strtok_r.o              \
563         strtoumax.o             \
564         swab.o                  \
565         swapctl.o               \
566         sysconf.o               \
567         syslog.o                \
568         tcdrain.o               \
569         tcflow.o                \
570         tcflush.o               \
571         tcgetattr.o             \
572         tcgetpgrp.o             \
573         tcgetsid.o              \
574         tcsendbreak.o           \
575         tcsetattr.o             \
576         tcsetpgrp.o             \
577         tell.o                  \
578         telldir.o               \
579         tfind.o                 \
580         time_data.o             \
581         time_gdata.o            \
582         tls_data.o              \
583         truncate.o              \
584         tsdalloc.o              \
585         tsearch.o               \
586         ttyname.o               \
587         ttyslot.o               \
588         ualarm.o                \
589         ucred.o                 \
```



```
590         valloc.o                \
591         vlfmt.o                 \
592         vpfmt.o                 \
593         waitpid.o               \
594         walkstack.o             \
595         wdata.o                 \
596         xgetwidth.o             \
597         xpg4.o                  \
598         xpg6.o

600 PORTPRINT_W=                     \
601         doprnt_w.o

603 PORTPRINT=                       \
604         asprintf.o              \
605         doprnt.o                \
606         fprintf.o               \
607         printf.o                \
608         snprintf.o              \
609         sprintf.o               \
610         vfprintf.o              \
611         vprintf.o               \
612         vsnprintf.o             \
613         vsprintf.o              \
614         vwprintf.o              \
615         wprintf.o

617 # c89 variants to support 32-bit size of c89 u/intmax_t (32-bit libc only)
618 PORTPRINT_C89=                  \
619         vfprintf_c89.o          \
620         vprintf_c89.o           \
621         vsnprintf_c89.o         \
622         vsprintf_c89.o          \
623         vwprintf_c89.o

625 PORTSTDIO_C89=                  \
626         vscanf_c89.o            \
627         vwscanf_c89.o           \

629 # portable stdio objects that contain large file interfaces.
630 # Note: fopen64 is a special case, as we build it small.
631 PORTSTDIO64=                    \
632         fopen64.o               \
633         fpos64.o

635 PORTSTDIO_W=                    \
636         doscan_w.o

638 PORTSTDIO=                      \
639         __extensions.o          \
640         _endopen.o              \
641         _filbuf.o               \
642         _findbuf.o              \
643         _flsbuf.o               \
644         _wrtchk.o               \
645         clearerr.o              \
646         ctermid.o               \
647         ctermid_r.o             \
648         cuserid.o               \
649         data.o                  \
650         doscan.o                \
651         fdopen.o                \
652         feof.o                  \
653         ferror.o                \
654         fgetc.o                 \
655         fgets.o                 \
```

```
656         fileno.o                \
657         flockf.o                \
658         flush.o                 \
659         fopen.o                 \
660         fpos.o                  \
661         fputc.o                 \
662         fputs.o                 \
663         fread.o                 \
664         fseek.o                 \
665         fseeko.o                \
666         ftell.o                 \
667         ftello.o                \
668         fwrite.o                \
669         getc.o                  \
670         getchar.o               \
671         getline.o               \
672         getpass.o               \
673         gets.o                  \
674         getw.o                  \
675         mse.o                   \
676         popen.o                 \
677         putc.o                  \
678         putchar.o               \
679         puts.o                  \
680         putw.o                  \
681         rewind.o                \
682         scanf.o                 \
683         setbuf.o                \
684         setbuffer.o             \
685         setvbuf.o               \
686         system.o                \
687         tempnam.o               \
688         tmpfile.o               \
689         tmpnam_r.o              \
690         ungetc.o                \
691         vscanf.o                \
692         vwscanf.o               \
693         wscanf.o

695 PORTI18N=                       \
696         getwchar.o              \
697         putwchar.o              \
698         putws.o                 \
699         strcasecmp.o            \
700         strcasestr.o            \
701         strncasecmp.o           \
702         strtows.o               \
703         wcsnlen.o               \
704         wcsstr.o                \
705         wcstoimax.o             \
706         wcstol.o                \
707         wcstoul.o               \
708         wcswcs.o                \
709         wmemchr.o               \
710         wmemcmp.o               \
711         wmemcpy.o               \
712         wmemmove.o              \
713         wmemset.o               \
714         wscasecmp.o             \
715         wscat.o                 \
716         wschr.o                 \
717         wscmp.o                 \
718         wscpy.o                 \
719         wscspn.o                \
720         wsdup.o                 \
721         wslen.o                 \
```



```
722         wsncasecmp.o            \
723         wsncat.o                \
724         wsncmp.o                \
725         wsncpy.o                \
726         wspbrk.o                \
727         wsprintf.o              \
728         wsrchr.o                \
729         wsscanf.o               \
730         wsspn.o                 \
731         wstod.o                 \
732         wstok.o                 \
733         wstol.o                 \
734         wstoll.o                \
735         wsxfrm.o                \
736         gettext.o               \
737         gettext_gnu.o           \
738         gettext_real.o          \
739         gettext_util.o          \
740         isdigit.o               \
741         plural_parser.o         \
742         wdresolve.o             \
743         _ctype.o                \
744         isascii.o               \
745         toascii.o

747 PORTI18N_COND=                  \
748         wcstol_longlong.o       \
749         wcstoul_longlong.o

751 PORTLOCALE=                    \
752         big5.o                  \
753         btowc.o                 \
754         collate.o               \
755         collcmp.o               \
756         euc.o                   \
757         fnmatch.o               \
758         fgetwc.o                \
759         fgetws.o                \
760         fix_grouping.o          \
761         fputwc.o                \
762         fputws.o                \
763         fwide.o                 \
764         gb18030.o               \
765         gb2312.o                \
766         gbk.o                   \
767         getdate.o               \
768         iswctype.o              \
769         ldpart.o                \
770         lmessages.o             \
771         lnumeric.o              \
772         lmonetary.o             \
773         localeconv.o            \
774         mbftowc.o               \
775         mblen.o                 \
776         mbrlen.o                \
777         mbrtowc.o               \
778         mbsinit.o               \
779         mbsnrtowcs.o            \
780         mbsrtowcs.o             \
781         mbstowcs.o              \
782         mbtowc.o                \
783         mskanji.o               \
784         nextwctype.o            \
785         nl_langinfo.o           \
786         none.o                  \
787         regcomp.o               \
```

```
788         regfree.o               \
789         regerror.o              \
790         regexec.o               \
791         rune.o                  \
792         runetype.o              \
793         setlocale.o             \
794         setrunelocale.o         \
795         strcoll.o               \
796         strfmon.o               \
797         strftime.o              \
798         strptime.o              \
799         strxfrm.o               \
800         table.o                 \
801         timelocal.o             \
802         tolower.o               \
803         towlower.o              \
804         ungetwc.o               \
805         utf8.o                  \
806         wcrtomb.o               \
807         wcscoll.o               \
808         wcsftime.o              \
809         wcsnrtombs.o            \
810         wcsrtombs.o             \
811         wcswidth.o              \
812         wcstombs.o              \
813         wcsxfrm.o               \
814         wctob.o                 \
815         wctomb.o                \
816         wctrans.o               \
817         wctype.o                \
818         wcwidth.o               \
819         wscol.o                 \
820         xlocale.o
819         wscol.o

822 AIOOBJS=                       \
823         aio.o                   \
824         aio_alloc.o             \
825         posix_aio.o

827 RTOBJS=                        \
828         clock_timer.o           \
829         mqueue.o                \
830         pos4obj.o               \
831         sched.o                 \
832         sem.o                   \
833         shm.o                   \
834         sigev_thread.o

836 TPOOLOBJS=                     \
837         thread_pool.o

839 THREADSOBJS=                   \
840         alloc.o                 \
841         assfail.o               \
842         cancel.o                \
843         door_calls.o            \
844         pthr_attr.o             \
845         pthr_barrier.o          \
846         pthr_cond.o             \
847         pthr_mutex.o            \
848         pthr_rwlock.o           \
849         pthread.o               \
850         rwlock.o                \
851         scalls.o                \
852         sema.o                  \
```



```
853         sigaction.o             \
854         spawn.o                 \
855         synch.o                 \
856         tdb_agent.o             \
857         thr.o                   \
858         thread_interface.o      \
859         tls.o                   \
860         tsd.o

862 THREADSMACHOBJS=               \
863         machdep.o

865 THREADSASMOBJS=                \
866         asm_subr.o

868 UNICODEOBJS=                   \
869         u8_textprep.o           \
870         uconv.o

872 UNWINDMACHOBJS=                \
873         unwind.o

875 UNWINDASMOBJS=                 \
876         unwind_frame.o

878 # objects that implement the transitional large file API
879 PORTSYS64=                     \
880         lockf64.o               \
881         stat64.o

883 PORTSYS=                       \
884         _autofssys.o            \
885         access.o                \
886         acctctl.o               \
887         bsd_signal.o            \
888         chmod.o                 \
889         chown.o                 \
890         corectl.o               \
891         exacctsys.o             \
892         execl.o                 \
893         execle.o                \
894         execv.o                 \
895         fcntl.o                 \
896         getpagesizes.o          \
897         getpeerucred.o          \
898         inst_sync.o             \
899         issetugid.o             \
900         label.o                 \
901         link.o                  \
902         lockf.o                 \
903         lwp.o                   \
904         lwp_cond.o              \
905         lwp_rwlock.o            \
906         lwp_sigmask.o           \
907         meminfosys.o            \
908         mkdir.o                 \
909         mknod.o                 \
910         msgsys.o                \
911         nfssys.o                \
912         open.o                  \
913         pgrpsys.o               \
914         posix_sigwait.o         \
915         ppriv.o                 \
916         psetsys.o               \
917         rctlsys.o               \
918         readlink.o              \
```

```
 919         rename.o                \
 920         sbrk.o                  \
 921         semsys.o                \
 922         set_errno.o             \
 923         sharefs.o               \
 924         shmsys.o                \
 925         sidsys.o                \
 926         siginterrupt.o          \
 927         signal.o                \
 928         sigpending.o            \
 929         sigstack.o              \
 930         stat.o                  \
 931         symlink.o               \
 932         tasksys.o               \
 933         time.o                  \
 934         time_util.o             \
 935         ucontext.o              \
 936         unlink.o                \
 937         ustat.o                 \
 938         utimesys.o              \
 939         zone.o

 941 PORTREGEX=                      \
 942         glob.o                  \
 943         regcmp.o                \
 944         regex.o                 \
 945         wordexp.o

 947 MOSTOBJS=                       \
 948         $(STRETS)               \
 949         $(CRTOBJS)              \
 950         $(DYNOBJS)              \
 951         $(FPOBJS)               \
 952         $(FPASMOBJS)            \
 953         $(ATOMICOBJS)           \
 954         $(XATTROBJS)            \
 955         $(COMOBJS)              \
 956         $(DTRACEOBJS)           \
 957         $(GENOBJS)              \
 958         $(PORTFP)               \
 959         $(PORTGEN)              \
 960         $(PORTGEN64)            \
 961         $(PORTI18N)             \
 962         $(PORTI18N_COND)        \
 963         $(PORTLOCALE)           \
 964         $(PORTPRINT)            \
 965         $(PORTPRINT_C89)        \
 966         $(PORTPRINT_W)          \
 967         $(PORTREGEX)            \
 968         $(PORTSTDIO)            \
 969         $(PORTSTDIO64)          \
 970         $(PORTSTDIO_C89)        \
 971         $(PORTSTDIO_W)          \
 972         $(PORTSYS)              \
 973         $(PORTSYS64)            \
 974         $(AIOOBJS)              \
 975         $(RTOBJS)               \
 976         $(TPOOLOBJS)            \
 977         $(THREADSOBJS)          \
 978         $(THREADSMACHOBJS)      \
 979         $(THREADSASMOBJS)       \
 980         $(UNICODEOBJS)          \
 981         $(UNWINDMACHOBJS)       \
 982         $(UNWINDASMOBJS)        \
 983         $(COMSYSOBJS)           \
 984         $(SYSOBJS)              \
```



```
 985         $(COMSYSOBJS64)         \
 986         $(SYSOBJS64)            \
 987         $(VALUES)

 989 TRACEOBJS=                      \
 990         plockstat.o

 992 # NOTE: libc.so.1 must be linked with the minimal crti.o and crtn.o
 993 # modules whose source is provided in the $(SRC)/lib/common directory.
 994 # This must be done because otherwise the Sun C compiler would insert
 995 # its own versions of these modules and those versions contain code
 996 # to call out to C++ initialization functions.  Such C++ initialization
 997 # functions can call back into libc before thread initialization is
 998 # complete and this leads to segmentation violations and other problems.
 999 # Since libc contains no C++ code, linking with the minimal crti.o and
1000 # crtn.o modules is safe and avoids the problems described above.
1001 OBJECTS= $(CRTI) $(MOSTOBJS) $(CRTN)
1002 CRTSRCS= ../../common/i386

1004 LDPASS_OFF=     $(POUND_SIGN)

1006 # include common library definitions
1007 include ../../Makefile.lib

1009 # we need to override the default SONAME here because we might
1010 # be building a variant object (still libc.so.1, but different filename)
1011 SONAME = libc.so.1

1013 CFLAGS += $(CCVERBOSE) $(CTF_FLAGS)

1015 # This is necessary to avoid problems with calling _ex_unwind().
1016 # We probably don't want any inlining anyway.
1017 XINLINE = -xinline=
1018 CFLAGS += $(XINLINE)

1020 CERRWARN += -_gcc=-Wno-parentheses
1021 CERRWARN += -_gcc=-Wno-switch
1022 CERRWARN += -_gcc=-Wno-uninitialized
1023 CERRWARN += -_gcc=-Wno-unused-value
1024 CERRWARN += -_gcc=-Wno-unused-label
1025 CERRWARN += -_gcc=-Wno-unused-variable
1026 CERRWARN += -_gcc=-Wno-type-limits
1027 CERRWARN += -_gcc=-Wno-char-subscripts
1028 CERRWARN += -_gcc=-Wno-clobbered
1029 CERRWARN += -_gcc=-Wno-unused-function
1030 CERRWARN += -_gcc=-Wno-address

1032 # Setting THREAD_DEBUG = -DTHREAD_DEBUG (make THREAD_DEBUG=-DTHREAD_DEBUG ...)
1033 # enables ASSERT() checking in the threads portion of the library.
1034 # This is automatically enabled for DEBUG builds, not for non-debug builds.
1035 THREAD_DEBUG =
1036 $(NOT_RELEASE_BUILD)THREAD_DEBUG = -DTHREAD_DEBUG

1038 # Make string literals read-only to save memory.
1039 CFLAGS += $(XSTRCONST)

1041 ALTPICS= $(TRACEOBJS:%=pics/%)

1043 $(DYNLIB) := BUILD.SO = $(LD) -o $@ -G $(DYNFLAGS) $(PICS) $(ALTPICS) \
1044                 $(EXTPICS) $(LDLIBS)

1046 MAPFILES =      $(LIBCDIR)/port/mapfile-vers

1048 #
1049 # EXTN_CPPFLAGS and EXTN_CFLAGS set in enclosing Makefile
1050 #
```

```
1051 CFLAGS +=       $(EXTN_CFLAGS)
1052 CPPFLAGS=       -D_REENTRANT -Di386 $(EXTN_CPPFLAGS) $(THREAD_DEBUG) \
1053                 -I$(LIBCBASE)/inc -I$(LIBCDIR)/inc $(CPPFLAGS.master)
1054 ASFLAGS=        $(AS_PICFLAGS) -P -D__STDC__ -D_ASM $(CPPFLAGS) $(i386_AS_XARCH)

1056 # As a favor to the dtrace syscall provider, libc still calls the
1057 # old syscall traps that have been obsoleted by the *at() interfaces.
1058 # Delete this to compile libc using only the new *at() system call traps
1059 CPPFLAGS += -D_RETAIN_OLD_SYSCALLS

1061 # Inform the run-time linker about libc specialized initialization
1062 RTLDINFO =      -z rtldinfo=tls_rtldinfo
1063 DYNFLAGS +=     $(RTLDINFO)

1065 # Force libc's internal references to be resolved immediately upon loading
1066 # in order to avoid critical region problems.  Since almost all libc symbols
1067 # are marked 'protected' in the mapfiles, this is a minimal set (15 to 20).
1068 DYNFLAGS +=     -znow

1070 DYNFLAGS +=     -e __rtboot
1071 DYNFLAGS +=     $(EXTN_DYNFLAGS)

1073 # Inform the kernel about the initial DTrace area (in case
1074 # libc is being used as the interpreter / runtime linker).
1075 DTRACE_DATA =   -zdtrace=dtrace_data
1076 DYNFLAGS +=     $(DTRACE_DATA)

1078 # DTrace needs an executable data segment.
1079 MAPFILE.NED=

1081 BUILD.s=        $(AS) $(ASFLAGS) $< -o $@

1083 # Override this top level flag so the compiler builds in its native
1084 # C99 mode.  This has been enabled to support the complex arithmetic
1085 # added to libc.
1086 C99MODE=        $(C99_ENABLE)

1088 # libc method of building an archive
1089 # The "$(GREP) -v ' L '" part is necessary only until
1090 # lorder is fixed to ignore thread-local variables.
1091 BUILD.AR= $(RM) $@ ; \
1092         $(AR) q $@ `$(LORDER) $(MOSTOBJS:%=$(DIR)/%) | $(GREP) -v ' L ' | $(TSOR

1094 # extra files for the clean target
1095 CLEANFILES=                     \
1096         $(LIBCDIR)/port/gen/errlst.c    \
1097         $(LIBCDIR)/port/gen/new_list.c  \
1098         assym.h                 \
1099         genassym                \
1100         crt/_rtld.s             \
1101         crt/_rtbootld.s         \
1102         pics/_rtbootld.o        \
1103         pics/crti.o             \
1104         pics/crtn.o             \
1105         $(ALTPICS)

1107 CLOBBERFILES += $(LIB_PIC)

1109 # list of C source for lint
1110 SRCS=                                                   \
1111         $(ATOMICOBJS:%.o=$(SRC)/common/atomic/%.c)      \
1112         $(XATTROBJS:%.o=$(SRC)/common/xattr/%.c)        \
1113         $(COMOBJS:%.o=$(SRC)/common/util/%.c)           \
1114         $(DTRACEOBJS:%.o=$(SRC)/common/dtrace/%.c)      \
1115         $(PORTFP:%.o=$(LIBCDIR)/port/fp/%.c)                    \
1116         $(PORTGEN:%.o=$(LIBCDIR)/port/gen/%.c)                  \
```



```
1117         $(PORTI18N:%.o=$(LIBCDIR)/port/i18n/%.c)                \
1118         $(PORTLOCALE:%.o=$(LIBCDIR)/port/locale/%.c)            \
1119         $(PORTPRINT:%.o=$(LIBCDIR)/port/print/%.c)              \
1120         $(PORTREGEX:%.o=$(LIBCDIR)/port/regex/%.c)              \
1121         $(PORTSTDIO:%.o=$(LIBCDIR)/port/stdio/%.c)              \
1122         $(PORTSYS:%.o=$(LIBCDIR)/port/sys/%.c)                  \
1123         $(AIOOBJS:%.o=$(LIBCDIR)/port/aio/%.c)                  \
1124         $(RTOBJS:%.o=$(LIBCDIR)/port/rt/%.c)                    \
1125         $(TPOOLOBJS:%.o=$(LIBCDIR)/port/tpool/%.c)              \
1126         $(THREADSOBJS:%.o=$(LIBCDIR)/port/threads/%.c)          \
1127         $(THREADSMACHOBJS:%.o=$(LIBCDIR)/$(MACH)/threads/%.c)   \
1128         $(UNICODEOBJS:%.o=$(SRC)/common/unicode/%.c)    \
1129         $(UNWINDMACHOBJS:%.o=$(LIBCDIR)/port/unwind/%.c)        \
1130         $(FPOBJS:%.o=$(LIBCDIR)/$(MACH)/fp/%.c)                 \
1131         $(LIBCBASE)/gen/ecvt.c                          \
1132         $(LIBCBASE)/gen/makectxt.c                      \
1133         $(LIBCBASE)/gen/siginfolst.c                    \
1134         $(LIBCBASE)/gen/siglongjmp.c                    \
1135         $(LIBCBASE)/gen/strcmp.c                        \
1136         $(LIBCBASE)/gen/sync_instruction_memory.c       \
1137         $(LIBCBASE)/sys/ptrace.c                        \
1138         $(LIBCBASE)/sys/uadmin.c

1140 # conditional assignments
1141 $(DYNLIB) := CRTI = crti.o
1142 $(DYNLIB) := CRTN = crtn.o

1144 # Files which need the threads .il inline template
1145 TIL=                            \
1146         aio.o                   \
1147         alloc.o                 \
1148         assfail.o               \
1149         atexit.o                \
1150         atfork.o                \
1151         cancel.o                \
1152         door_calls.o            \
1153         err.o                   \
1154         errno.o                 \
1155         lwp.o                   \
1156         ma.o                    \
1157         machdep.o               \
1158         posix_aio.o             \
1159         pthr_attr.o             \
1160         pthr_barrier.o          \
1161         pthr_cond.o             \
1162         pthr_mutex.o            \
1163         pthr_rwlock.o           \
1164         pthread.o               \
1165         rand.o                  \
1166         rwlock.o                \
1167         scalls.o                \
1168         sched.o                 \
1169         sema.o                  \
1170         sigaction.o             \
1171         sigev_thread.o          \
1172         spawn.o                 \
1173         stack.o                 \
1174         synch.o                 \
1175         tdb_agent.o             \
1176         thr.o                   \
1177         thread_interface.o      \
1178         thread_pool.o           \
1179         tls.o                   \
1180         tsd.o                   \
1181         unwind.o
```

```
1183 THREADS_INLINES = $(LIBCBASE)/threads/i386.il
1184 $(TIL:%=pics/%) := CFLAGS += $(THREADS_INLINES)

1186 # pics/mul64.o := CFLAGS += $(LIBCBASE)/crt/mul64.il

1188 # large-file-aware components that should be built large

1190 $(COMSYSOBJS64:%=pics/%) := \
1191         CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1193 $(SYSOBJS64:%=pics/%) := \
1194         CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1196 $(PORTGEN64:%=pics/%) := \
1197         CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1199 $(PORTSTDIO64:%=pics/%) := \
1200         CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1202 $(PORTSYS64:%=pics/%) := \
1203         CPPFLAGS += -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64

1205 $(PORTSTDIO_W:%=pics/%) := \
1206         CPPFLAGS += -D_WIDE

1208 $(PORTPRINT_W:%=pics/%) := \
1209         CPPFLAGS += -D_WIDE

1211 $(PORTPRINT_C89:%=pics/%) := \
1212         CPPFLAGS += -D_C89_INTMAX32

1214 $(PORTSTDIO_C89:%=pics/%) := \
1215         CPPFLAGS += -D_C89_INTMAX32

1217 $(PORTI18N_COND:%=pics/%) := \
1218         CPPFLAGS += -D_WCS_LONGLONG

1220 .KEEP_STATE:

1222 all: $(LIBS) $(LIB_PIC)

1224 lint    :=      CPPFLAGS += -I$(LIBCDIR)/$(MACH)/fp
1225 lint    :=      CPPFLAGS += -D_MSE_INT_H -D_LCONV_C99
1226 lint    :=      LINTFLAGS += -mn -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED

1228 lint:
1229         @echo $(LINT.c) ...
1230         @$(LINT.c) $(SRCS) $(LDLIBS)

1232 $(LINTLIB):= SRCS=$(LIBCDIR)/port/llib-lc
1233 $(LINTLIB):= CPPFLAGS += -D_MSE_INT_H
1234 $(LINTLIB):= LINTFLAGS=-nvx

1236 # object files that depend on inline template
1237 $(TIL:%=pics/%): $(LIBCBASE)/threads/i386.il
1238 # pics/mul64.o: $(LIBCBASE)/crt/mul64.il

1240 # include common libc targets
1241 include $(LIBCDIR)/Makefile.targ

1243 # We need to strip out all CTF and DOF data from the static library
1244 $(LIB_PIC) := DIR = pics
1245 $(LIB_PIC): pics $$(PICS)
1246         $(BUILD.AR)
1247         $(MCS) -d -n .SUNW_ctf $@ > /dev/null 2>&1
1248         $(MCS) -d -n .SUNW_dof $@ > /dev/null 2>&1
```



```
1249         $(AR) -ts $@ > /dev/null
1250         $(POST_PROCESS_A)

1252 $(LIBCBASE)/crt/_rtbootld.s: $(LIBCBASE)/crt/_rtboot.s $(LIBCBASE)/crt/_rtld.c
1253         $(CC) $(CPPFLAGS) $(CTF_FLAGS) -O -S $(C_PICFLAGS) \
1254             $(LIBCBASE)/crt/_rtld.c -o $(LIBCBASE)/crt/_rtld.s
1255         $(CAT) $(LIBCBASE)/crt/_rtboot.s $(LIBCBASE)/crt/_rtld.s > $@
1256         $(RM) $(LIBCBASE)/crt/_rtld.s

1258 # partially built from C source
1259 pics/_rtbootld.o: $(LIBCBASE)/crt/_rtbootld.s
1260         $(AS) $(ASFLAGS) $(LIBCBASE)/crt/_rtbootld.s -o $@
1261         $(CTFCONVERT_O)

1263 ASSYMDEP_OBJS=                  \
1264         _lwp_mutex_unlock.o     \
1265         _stack_grow.o           \
1266         getcontext.o            \
1267         setjmp.o                \
1268         tls_get_addr.o          \
1269         vforkx.o

1271 $(ASSYMDEP_OBJS:%=pics/%)       :=      CPPFLAGS += -I.

1273 $(ASSYMDEP_OBJS:%=pics/%): assym.h

1275 # assym.h build rules

1277 GENASSYM_C = $(LIBCDIR)/$(MACH)/genassym.c

1279 genassym: $(GENASSYM_C)
1280         $(NATIVECC) -I$(LIBCBASE)/inc -I$(LIBCDIR)/inc  \
1281                 -D__EXTENSIONS__ $(CPPFLAGS.native) -o $@ $(GENASSYM_C)

1283 OFFSETS = $(LIBCDIR)/$(MACH)/offsets.in

1285 assym.h: $(OFFSETS) genassym
1286         $(OFFSETS_CREATE) <$(OFFSETS) >$@
1287         ./genassym >>$@

1289 # derived C source and related explicit dependencies
1290 $(LIBCDIR)/port/gen/errlst.c + \
1291 $(LIBCDIR)/port/gen/new_list.c: $(LIBCDIR)/port/gen/errlist $(LIBCDIR)/port/gen/
1292         cd $(LIBCDIR)/port/gen; pwd; $(AWK) -f errlist.awk < errlist

1294 pics/errlst.o: $(LIBCDIR)/port/gen/errlst.c

1296 pics/new_list.o: $(LIBCDIR)/port/gen/new_list.c
```

```
**********************************************************
   14587 Thu Jul 11 12:26:11 2013
new/usr/src/lib/libc/port/locale/collate.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1995 Alex Tatmanjants <alex@elvisti.kiev.ua>
 *              at Electronni Visti IA, Kiev, Ukraine.
 *                      All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include "file64.h"
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <wchar.h>
#include <errno.h>
#include <unistd.h>
#include <ctype.h>
#include <unistd.h>
#include <fcntl.h>
#include <assert.h>
#include <sys/stat.h>
#include <sys/mman.h>

#include "collate.h"
#include "setlocale.h"
#include "ldpart.h"

/*
 * See the comments in usr/src/cmd/localedef/collate.c for further
 * information.  It would also be very helpful to have a copy of the
 * POSIX standard for collation (in the locale format manual page)
 * handy (www.opengroup.org).
 */

static collate_subst_t          *subst_table[COLL_WEIGHTS_MAX];
static collate_char_t           *char_pri_table;
static collate_large_t          *large_pri_table;
static collate_chain_t          *chain_pri_table;
static char                     *cache = NULL;
static size_t                   cachesz;
```



```
static char                     collate_encoding[ENCODING_LEN + 1];

/* Exposed externally to other parts of libc. */
collate_info_t                  *_collate_info;
int _collate_load_error = 1;

struct xlocale_collate __xlocale_global_collate = {
        {{0}, "C"}, 1, 0
};

struct xlocale_collate __xlocale_C_collate = {
        {{0}, "C"}, 1, 0
};

static void
destruct_collate(void *t)
{
        struct xlocale_collate *table = t;

        /* XXX */;
}

void *
__collate_load(const char *encoding, locale_t unused)
{
        struct xlocale_collate *table;

        if (strcmp(encoding, "C") == 0 || strcmp(encoding, "POSIX") == 0) {
                return &__xlocale_C_collate;
        }

        table = calloc(sizeof(struct xlocale_collate), 1);
        if (table == NULL) {
                /* XXX */
        }

        return (table);
}
#endif /* ! codereview */

int
_collate_load_tables(const char *encoding)
{
        int i, chains, z;
        char buf[PATH_MAX];
        char *TMP;
        char *map;
        collate_info_t *info;
        struct stat sbuf;
        int fd;

        /* 'encoding' must be already checked. */
        if (strcmp(encoding, "C") == 0 || strcmp(encoding, "POSIX") == 0) {
                _collate_load_error = 1;
                return (_LDP_CACHE);
        }

        /*
         * If the locale name is the same as our cache, use the cache.
         */
        if (cache && (strncmp(encoding, collate_encoding, ENCODING_LEN) == 0)) {
                _collate_load_error = 0;
                return (_LDP_CACHE);
        }

        /*
```

```
128          * Slurp the locale file into the cache.
129          */
130
131         (void) snprintf(buf, sizeof (buf), "%s/%s/LC_COLLATE/LCL_DATA",
132             _PathLocale, encoding);
133
134         if ((fd = open(buf, O_RDONLY)) < 0)
135                 return (_LDP_ERROR);
136         if (fstat(fd, &sbuf) < 0) {
137                 (void) close(fd);
138                 return (_LDP_ERROR);
139         }
140         if (sbuf.st_size < (COLLATE_STR_LEN + sizeof (info))) {
141                 (void) close(fd);
142                 errno = EINVAL;
143                 return (_LDP_ERROR);
144         }
145         map = mmap(NULL, sbuf.st_size, PROT_READ, MAP_PRIVATE, fd, 0);
146         (void) close(fd);
147         if ((TMP = map) == NULL) {
148                 return (_LDP_ERROR);
149         }
150
151         if (strncmp(TMP, COLLATE_VERSION, COLLATE_STR_LEN) != 0) {
152                 (void) munmap(map, sbuf.st_size);
153                 errno = EINVAL;
154                 return (_LDP_ERROR);
155         }
156         TMP += COLLATE_STR_LEN;
157
158         info = (void *)TMP;
159         TMP += sizeof (*info);
160
161         if ((info->directive_count < 1) ||
162             (info->directive_count >= COLL_WEIGHTS_MAX) ||
163             ((chains = info->chain_count) < 0)) {
164                 (void) munmap(map, sbuf.st_size);
165                 errno = EINVAL;
166                 return (_LDP_ERROR);
167         }
168
169         i = (sizeof (collate_char_t) * (UCHAR_MAX + 1)) +
170             (sizeof (collate_chain_t) * chains) +
171             (sizeof (collate_large_t) * info->large_count);
172         for (z = 0; z < (info->directive_count); z++) {
173                 i += sizeof (collate_subst_t) * info->subst_count[z];
174         }
175         if (i != (sbuf.st_size - (TMP - map))) {
176                 (void) munmap(map, sbuf.st_size);
177                 errno = EINVAL;
178                 return (_LDP_ERROR);
179         }
180
181         char_pri_table = (void *)TMP;
182         TMP += sizeof (collate_char_t) * (UCHAR_MAX + 1);
183
184         for (z = 0; z < info->directive_count; z++) {
185                 if (info->subst_count[z] > 0) {
186                         subst_table[z] = (void *)TMP;
187                         TMP += info->subst_count[z] * sizeof (collate_subst_t);
188                 } else {
189                         subst_table[z] = NULL;
190                 }
191         }
192
193         if (chains > 0) {
```



```
194                 chain_pri_table = (void *)TMP;
195                 TMP += chains * sizeof (collate_chain_t);
196         } else
197                 chain_pri_table = NULL;
198         if (info->large_count > 0)
199                 large_pri_table = (void *)TMP;
200         else
201                 large_pri_table = NULL;
202
203         (void) strlcpy(collate_encoding, encoding, ENCODING_LEN);
204         _collate_info = info;
205
206         if (cache)
207                 (void) munmap(cache, cachesz);
208
209         cache = map;
210         cachesz = sbuf.st_size;
211         _collate_load_error = 0;
212
213         return (_LDP_LOADED);
214 }
215
216 static int32_t *
217 substsearch(const wchar_t key, int pass)
218 {
219         collate_subst_t *p;
220         int n = _collate_info->subst_count[pass];
221
222         if (n == 0)
223                 return (NULL);
224
225         if (pass >= _collate_info->directive_count)
226                 return (NULL);
227
228         if (!(key & COLLATE_SUBST_PRIORITY))
229                 return (NULL);
230
231         p = subst_table[pass] + (key & ~COLLATE_SUBST_PRIORITY);
232         assert(p->key == key);
233         return (p->pri);
234 }
235
236 /*
237  * Note: for performance reasons, we have expanded bsearch here.  This avoids
238  * function call overhead with each comparison.
239  */
240
241 static collate_chain_t *
242 chainsearch(const wchar_t *key, int *len)
243 {
244         int low;
245         int high;
246         int next, compar, l;
247         collate_chain_t *p;
248         collate_chain_t *tab;
249
250         if (_collate_info->chain_count == 0)
251                 return (NULL);
252
253         low = 0;
254         high = _collate_info->chain_count - 1;
255         tab = chain_pri_table;
256
257         while (low <= high) {
258                 next = (low + high) / 2;
259                 p = tab + next;
```

```
260                 compar = *key - *p->str;
261                 if (compar == 0) {
262                         l = wcsnlen(p->str, COLLATE_STR_LEN);
263                         compar = wcsncmp(key, p->str, l);
264                         if (compar == 0) {
265                                 *len = l;
266                                 return (p);
267                         }
268                 }
269                 if (compar > 0)
270                         low = next + 1;
271                 else
272                         high = next - 1;
273         }
274         return (NULL);
275 }
277 static collate_large_t *
278 largesearch(const wchar_t key)
279 {
280         int low = 0;
281         int high = _collate_info->large_count - 1;
282         int next, compar;
283         collate_large_t *p;
284         collate_large_t *tab = large_pri_table;
286         if (_collate_info->large_count == 0)
287                 return (NULL);
289         while (low <= high) {
290                 next = (low + high) / 2;
291                 p = tab + next;
292                 compar = key - p->val;
293                 if (compar == 0)
294                         return (p);
295                 if (compar > 0)
296                         low = next + 1;
297                 else
298                         high = next - 1;
299         }
300         return (NULL);
301 }
303 void
304 _collate_lookup(const wchar_t *t, int *len, int *pri, int which, int **state)
305 {
306         collate_chain_t *p2;
307         collate_large_t *match;
308         collate_info_t *info = _collate_info;
309         int p, l;
310         int *sptr;
312         /*
313          * If this is the "last" pass for the UNDEFINED, then
314          * we just return the priority itself.
315          */
316         if (which >= info->directive_count) {
317                 *pri = *t;
318                 *len = 1;
319                 *state = NULL;
320                 return;
321         }
323         /*
324          * If we have remaining substitution data from a previous
325          * call, consume it first.
```



```
326          */
327         if ((sptr = *state) != NULL) {
328                 *pri = *sptr;
329                 sptr++;
330                 *state = *sptr ? sptr : NULL;
331                 *len = 0;
332                 return;
333         }
335         /* No active substitutions */
336         *len = 1;
338         /*
339          * Check for composites such as dipthongs that collate as a
340          * single element (aka chains or collating-elements).
341          */
342         if (((p2 = chainsearch(t, &l)) != NULL) &&
343             ((p = p2->pri[which]) >= 0)) {
345                 *len = l;
346                 *pri = p;
348         } else if (*t <= UCHAR_MAX) {
350                 /*
351                  * Character is a small (8-bit) character.
352                  * We just look these up directly for speed.
353                  */
354                 *pri = char_pri_table[*t].pri[which];
356         } else if ((info->large_count > 0) &&
357             ((match = largesearch(*t)) != NULL)) {
359                 /*
360                  * Character was found in the extended table.
361                  */
362                 *pri = match->pri.pri[which];
364         } else {
365                 /*
366                  * Character lacks a specific definition.
367                  */
368                 if (info->directive[which] & DIRECTIVE_UNDEFINED) {
369                         /* Mask off sign bit to prevent ordering confusion. */
370                         *pri = (*t & COLLATE_MAX_PRIORITY);
371                 } else {
372                         *pri = info->undef_pri[which];
373                 }
374                 /* No substitutions for undefined characters! */
375                 return;
376         }
378         /*
379          * Try substituting (expanding) the character.  We are
380          * currently doing this *after* the chain compression.  I
381          * think it should not matter, but this way might be slightly
382          * faster.
383          *
384          * We do this after the priority search, as this will help us
385          * to identify a single key value.  In order for this to work,
386          * its important that the priority assigned to a given element
387          * to be substituted be unique for that level.  The localedef
388          * code ensures this for us.
389          */
390         if ((sptr = substsearch(*pri, which)) != NULL) {
391                 if ((*pri = *sptr) != 0) {
```

```
392                         sptr++;
393                         *state = *sptr ? sptr : NULL;
394                 }
395         }

397 }

399 /*
400  * This is the meaty part of wcsxfrm & strxfrm.  Note that it does
401  * NOT NULL terminate.  That is left to the caller.
402  */
403 size_t
404 _collate_wxfrm(const wchar_t *src, wchar_t *xf, size_t room)
405 {
406         int             pri;
407         int             len;
408         const wchar_t   *t;
409         wchar_t         *tr = NULL;
410         int             direc;
411         int             pass;
412         int32_t         *state;
413         size_t          want = 0;
414         size_t          need = 0;

416         assert(src);

418         for (pass = 0; pass <= _collate_info->directive_count; pass++) {

420                 state = NULL;

422                 if (pass != 0) {
423                         /* insert level separator from the previous pass */
424                         if (room) {
425                                 *xf++ = 1;
426                                 room--;
427                         }
428                         want++;
429                 }

431                 /* special pass for undefined */
432                 if (pass == _collate_info->directive_count) {
433                         direc = DIRECTIVE_FORWARD | DIRECTIVE_UNDEFINED;
434                 } else {
435                         direc = _collate_info->directive[pass];
436                 }

438                 t = src;

440                 if (direc & DIRECTIVE_BACKWARD) {
441                         wchar_t *bp, *fp, c;
442                         if (tr)
443                                 free(tr);
444                         if ((tr = wcsdup(t)) == NULL) {
445                                 errno = ENOMEM;
446                                 goto fail;
447                         }
448                         bp = tr;
449                         fp = tr + wcslen(tr) - 1;
450                         while (bp < fp) {
451                                 c = *bp;
452                                 *bp++ = *fp;
453                                 *fp-- = c;
454                         }
455                         t = (const wchar_t *)tr;
456                 }
```



```
458                 if (direc & DIRECTIVE_POSITION) {
459                         while (*t || state) {
460                                 _collate_lookup(t, &len, &pri, pass, &state);
461                                 t += len;
462                                 if (pri <= 0) {
463                                         if (pri < 0) {
464                                                 errno = EINVAL;
465                                                 goto fail;
466                                         }
467                                         pri = COLLATE_MAX_PRIORITY;
468                                 }
469                                 if (room) {
470                                         *xf++ = pri;
471                                         room--;
472                                 }
473                                 want++;
474                                 need = want;
475                         }
476                 } else {
477                         while (*t || state) {
478                                 _collate_lookup(t, &len, &pri, pass, &state);
479                                 t += len;
480                                 if (pri <= 0) {
481                                         if (pri < 0) {
482                                                 errno = EINVAL;
483                                                 goto fail;
484                                         }
485                                         continue;
486                                 }
487                                 if (room) {
488                                         *xf++ = pri;
489                                         room--;
490                                 }
491                                 want++;
492                                 need = want;
493                         }
494                 }
495         }

497 end:
498         if (tr)
499                 free(tr);
500         return (need);

502 fail:
503         if (tr)
504                 free(tr);
505         return ((size_t)(-1));
506 }

508 /*
509  * In the non-POSIX case, we transform each character into a string of
510  * characters representing the character's priority.  Since char is usually
511  * signed, we are limited by 7 bits per byte.  To avoid zero, we need to add
512  * XFRM_OFFSET, so we can't use a full 7 bits.  For simplicity, we choose 6
513  * bits per byte.
514  *
515  * It turns out that we sometimes have real priorities that are
516  * 31-bits wide.  (But: be careful using priorities where the high
517  * order bit is set -- i.e. the priority is negative.  The sort order
518  * may be surprising!)
519  *
520  * TODO: This would be a good area to optimize somewhat.  It turns out
521  * that real prioririties *except for the last UNDEFINED pass* are generally
522  * very small.  We need the localedef code to precalculate the max
523  * priority for us, and ideally also give us a mask, and then we could
```

```
524  * severely limit what we expand to.
525  */
526 #define XFRM_BYTES      6
527 #define XFRM_OFFSET     ('0')   /* make all printable characters */
528 #define XFRM_SHIFT      6
529 #define XFRM_MASK       ((1 << XFRM_SHIFT) - 1)
530 #define XFRM_SEP        ('.')   /* chosen to be less than XFRM_OFFSET */

532 static int
533 xfrm(unsigned char *p, int pri, int pass)
534 {
535         /* we use unsigned to ensure zero fill on right shift */
536         uint32_t val = (uint32_t)_collate_info->pri_count[pass];
537         int nc = 0;

539         while (val) {
540                 *p = (pri & XFRM_MASK) + XFRM_OFFSET;
541                 pri >>= XFRM_SHIFT;
542                 val >>= XFRM_SHIFT;
543                 p++;
544                 nc++;
545         }
546         return (nc);
547 }

549 size_t
550 _collate_sxfrm(const wchar_t *src, char *xf, size_t room)
551 {
552         int             pri;
553         int             len;
554         const wchar_t   *t;
555         wchar_t         *tr = NULL;
556         int             direc;
557         int             pass;
558         int32_t         *state;
559         size_t          want = 0;
560         size_t          need = 0;
561         int             b;
562         uint8_t         buf[XFRM_BYTES];

564         assert(src);

566         for (pass = 0; pass <= _collate_info->directive_count; pass++) {

568                 state = NULL;

570                 if (pass != 0) {
571                         /* insert level separator from the previous pass */
572                         if (room) {
573                                 *xf++ = XFRM_SEP;
574                                 room--;
575                         }
576                         want++;
577                 }

579                 /* special pass for undefined */
580                 if (pass == _collate_info->directive_count) {
581                         direc = DIRECTIVE_FORWARD | DIRECTIVE_UNDEFINED;
582                 } else {
583                         direc = _collate_info->directive[pass];
584                 }

586                 t = src;

588                 if (direc & DIRECTIVE_BACKWARD) {
589                         wchar_t *bp, *fp, c;
```



```
590                         if (tr)
591                                 free(tr);
592                         if ((tr = wcsdup(t)) == NULL) {
593                                 errno = ENOMEM;
594                                 goto fail;
595                         }
596                         bp = tr;
597                         fp = tr + wcslen(tr) - 1;
598                         while (bp < fp) {
599                                 c = *bp;
600                                 *bp++ = *fp;
601                                 *fp-- = c;
602                         }
603                         t = (const wchar_t *)tr;
604                 }

606                 if (direc & DIRECTIVE_POSITION) {
607                         while (*t || state) {

609                                 _collate_lookup(t, &len, &pri, pass, &state);
610                                 t += len;
611                                 if (pri <= 0) {
612                                         if (pri < 0) {
613                                                 errno = EINVAL;
614                                                 goto fail;
615                                         }
616                                         pri = COLLATE_MAX_PRIORITY;
617                                 }

619                                 b = xfrm(buf, pri, pass);
620                                 want += b;
621                                 if (room) {
622                                         while (b) {
623                                                 b--;
624                                                 if (room) {
625                                                         *xf++ = buf[b];
626                                                         room--;
627                                                 }
628                                         }
629                                 }
630                                 need = want;
631                         }
632                 } else {
633                         while (*t || state) {
634                                 _collate_lookup(t, &len, &pri, pass, &state);
635                                 t += len;
636                                 if (pri <= 0) {
637                                         if (pri < 0) {
638                                                 errno = EINVAL;
639                                                 goto fail;
640                                         }
641                                         continue;
642                                 }

644                                 b = xfrm(buf, pri, pass);
645                                 want += b;
646                                 if (room) {

648                                         while (b) {
649                                                 b--;
650                                                 if (room) {
651                                                         *xf++ = buf[b];
652                                                         room--;
653                                                 }
654                                         }
655                                 }
```

```
656                                 need = want;
657                         }
658                 }
659         }

661 end:
662         if (tr)
663                 free(tr);
664         return (need);

666 fail:
667         if (tr)
668                 free(tr);
669         return ((size_t)(-1));
670 }
```

```
**********************************************************
    3964 Thu Jul 11 12:26:11 2013
new/usr/src/lib/libc/port/locale/collate.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systmes, Inc.  All rights reserved.
 * Copyright (c) 1995 Alex Tatmanjants <alex@elvisti.kiev.ua>
 *              at Electronni Visti IA, Kiev, Ukraine.
 *                      All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _COLLATE_H_
#define _COLLATE_H_

#include <sys/types.h>
#include <limits.h>

#include "xlocale_private.h"

#endif /* ! codereview */
#define COLLATE_STR_LEN         24              /* should be 64-bit multiple */
#define COLLATE_VERSION         "IllumosCollate2\n"

#define COLLATE_MAX_PRIORITY    (0x7fffffff)    /* max signed value */
#define COLLATE_SUBST_PRIORITY  (0x40000000)    /* bit indicates subst table */

#define DIRECTIVE_UNDEF         0x00
#define DIRECTIVE_FORWARD       0x01
#define DIRECTIVE_BACKWARD      0x02
#define DIRECTIVE_POSITION      0x04
#define DIRECTIVE_UNDEFINED     0x08    /* special last weight for UNDEFINED */

#define DIRECTIVE_DIRECTION_MASK (DIRECTIVE_FORWARD | DIRECTIVE_BACKWARD)

/*
 * The collate file format is as follows:
 *
 * char         version[COLLATE_STR_LEN];       // must be COLLATE_VERSION
 * collate_info_t       info;                   // see below, includes padding
 * collate_char_pri_t   char_data[256];         // 8 bit char values
 * collate_subst_t      subst[*];               // 0 or more substitutions
 * collate_chain_pri_t  chains[*];              // 0 or more chains
 * collate_large_pri_t  large[*];               // extended char priorities
 *
```



```
 * Note that all structures must be 32-bit aligned, as each structure
 * contains 32-bit member fields.  The entire file is mmap'd, so its
 * critical that alignment be observed.  It is not generally safe to
 * use any 64-bit values in the structures.
 */

typedef struct collate_info {
        uint8_t directive_count;
        uint8_t directive[COLL_WEIGHTS_MAX];
        int32_t pri_count[COLL_WEIGHTS_MAX];
        int32_t flags;
        int32_t chain_count;
        int32_t large_count;
        int32_t subst_count[COLL_WEIGHTS_MAX];
        int32_t undef_pri[COLL_WEIGHTS_MAX];
} collate_info_t;

typedef struct collate_char {
        int32_t pri[COLL_WEIGHTS_MAX];
} collate_char_t;

typedef struct collate_chain {
        wchar_t str[COLLATE_STR_LEN];
        int32_t pri[COLL_WEIGHTS_MAX];
} collate_chain_t;

typedef struct collate_large {
        int32_t val;
        collate_char_t pri;
} collate_large_t;

typedef struct collate_subst {
        int32_t key;
        int32_t pri[COLLATE_STR_LEN];
} collate_subst_t;

struct xlocale_collate {
        struct xlocale_component header;
        int __collate_load_error;
        int __collate_substitute_nontrivial;

        /* XXX */
};

#endif /* ! codereview */
int     _collate_load_tables(const char *);
void    _collate_lookup(const wchar_t *, int *, int *, int, int **);
size_t  _collate_wxfrm(const wchar_t *, wchar_t *, size_t);
size_t  _collate_sxfrm(const wchar_t *, char *, size_t);
int     _collate_range_cmp(wchar_t, wchar_t);

extern int _collate_load_error;
extern int _collate_substitute_nontrivial;
extern collate_info_t *_collate_info;

#endif /* !_COLLATE_H_ */
```

```
**********************************************************
    3459 Thu Jul 11 12:26:11 2013
new/usr/src/lib/libc/port/locale/lmessages.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <stddef.h>
#include <stdlib.h>
#endif /* ! codereview */
#include "ldpart.h"
#include "lmessages.h"

#define LCMESSAGES_SIZE_FULL (sizeof (struct lc_messages_T) / sizeof (char *))
#define LCMESSAGES_SIZE_MIN \
        (offsetof(struct lc_messages_T, yesstr) / sizeof (char *))

struct xlocale_messages __xlocale_global_messages;

#endif /* ! codereview */
static char empty[] = "";

static const struct lc_messages_T _C_messages_locale = {
        "^[yY]",        /* yesexpr */
        "^[nN]",        /* noexpr */
        "yes",          /* yesstr */
        "no"            /* nostr */
};

static void
destruct_messages(void *v)
{
        struct xlocale_messages *l = v;
```



```
        if (l->buffer != NULL)
                free(l->buffer);

        free(l);
}
static struct lc_messages_T _messages_locale;
static int      _messages_using_locale;
static char     *_messages_locale_buf;

static int
messages_load_locale(struct xlocale_messages *loc, int *using_locale,
    const char *name)
int
__messages_load_locale(const char *name)
{
        int ret;
        struct lc_messages_T *l = &loc->locale;
#endif /* ! codereview */

        ret = __part_load_locale(name, using_locale,
            &loc->buffer, "LC_MESSAGES",
        ret = __part_load_locale(name, &_messages_using_locale,
            &_messages_locale_buf, "LC_MESSAGES",
            LCMESSAGES_SIZE_FULL, LCMESSAGES_SIZE_MIN,
            (const char **)l);
            (const char **)&_messages_locale);
        if (ret == _LDP_LOADED) {
                if (l->yesstr == NULL)
                        l->yesstr = empty;
                if (l->nostr == NULL)
                        l->nostr = empty;
                if (_messages_locale.yesstr == NULL)
                        _messages_locale.yesstr = empty;
                if (_messages_locale.nostr == NULL)
                        _messages_locale.nostr = empty;
        }

#endif /* ! codereview */
        return (ret);
}

int
__messages_load_locale(const char *name)
{
        return (messages_load_locale(&__xlocale_global_messages,
            &__xlocale_global_locale.using_messages_locale, name));
}

void *
__messages_load(const char *name, locale_t l)
{
        struct xlocale_messages *new;

        new = calloc(sizeof(struct xlocale_messages), 1);
        /* XXX */
        new->header.header.destructor = destruct_messages;
        if (messages_load_locale(new, &l->using_messages_locale, name) ==
            _LDP_ERROR) {
                xlocale_release(new);
                return (NULL);
        }

        return (new);
}

#endif /* ! codereview */
```

```
116 struct lc_messages_T *
117 __get_current_messages_locale(locale_t loc)
 24 __get_current_messages_locale(void)
118 {
119         return (loc->using_messages_locale
120             ? &((struct xlocale_messages *)loc->components[XLC_MESSAGES])->local
121             : (struct lc_messages_T *)&_C_messages_locale);
 26         return (_messages_using_locale ? &_messages_locale :
 27             (struct lc_messages_T *)&_C_messages_locale);
122 }
______unchanged_portion_omitted_
```

```
**********************************************************
    1996 Thu Jul 11 12:26:11 2013
new/usr/src/lib/libc/port/locale/lmessages.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
 1 /*
 2  * Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
 3  * All rights reserved.
 4  *
 5  * Copyright (c) 2011 The FreeBSD Foundation
 6  * All rights reserved.
 7  * Portions of this software were developed by David Chisnall
 8  * under sponsorship from the FreeBSD Foundation.
 9  *
10 #endif /* ! codereview */
11  * Redistribution and use in source and binary forms, with or without
12  * modification, are permitted provided that the following conditions
13  * are met:
14  * 1. Redistributions of source code must retain the above copyright
15  *    notice, this list of conditions and the following disclaimer.
16  * 2. Redistributions in binary form must reproduce the above copyright
17  *    notice, this list of conditions and the following disclaimer in the
18  *    documentation and/or other materials provided with the distribution.
19  *
20  * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
21  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
22  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
23  * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
24  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
25  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
26  * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
27  * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
28  * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
29  * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
30  * SUCH DAMAGE.
31  */

33 #ifndef _LMESSAGES_H_
34 #define _LMESSAGES_H_

36 #include "xlocale_private.h"

38 #endif /* ! codereview */
39 struct  lc_messages_T {
40         const char      *yesexpr;
41         const char      *noexpr;
42         const char      *yesstr;
43         const char      *nostr;
44 };

46 struct xlocale_messages {
47         struct xlocale_component header;
48         char *buffer;
49         struct lc_messages_T locale;
50 };

52 struct lc_messages_T *__get_current_messages_locale(locale_t);
 5 struct lc_messages_T *__get_current_messages_locale(void);
53 int     __messages_load_locale(const char *);

55 #endif /* !_LMESSAGES_H_ */
```

```
**********************************************************
    5464 Thu Jul 11 12:26:12 2013
new/usr/src/lib/libc/port/locale/lmonetary.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <limits.h>
#include <stddef.h>
#include <stdlib.h>

#endif /* ! codereview */
#include "ldpart.h"
#include "lmonetary.h"

extern int __mlocale_changed;
extern const char *__fix_locale_grouping_str(const char *);

#define LCMONETARY_SIZE_FULL (sizeof (struct lc_monetary_T) / sizeof (char *))
#define LCMONETARY_SIZE_MIN \
        (offsetof(struct lc_monetary_T, int_p_cs_precedes) / sizeof (char *))

static char     empty[] = "";
static char     numempty[] = { CHAR_MAX, '\0' };

static const struct lc_monetary_T _C_monetary_locale = {
        empty,          /* int_curr_symbol */
        empty,          /* currency_symbol */
        empty,          /* mon_decimal_point */
        empty,          /* mon_thousands_sep */
        numempty,       /* mon_grouping */
        empty,          /* positive_sign */
        empty,          /* negative_sign */
        numempty,       /* int_frac_digits */
```



```
        numempty,       /* frac_digits */
        numempty,       /* p_cs_precedes */
        numempty,       /* p_sep_by_space */
        numempty,       /* n_cs_precedes */
        numempty,       /* n_sep_by_space */
        numempty,       /* p_sign_posn */
        numempty,       /* n_sign_posn */
        numempty,       /* int_p_cs_precedes */
        numempty,       /* int_n_cs_precedes */
        numempty,       /* int_p_sep_by_space */
        numempty,       /* int_n_sep_by_space */
        numempty,       /* int_p_sign_posn */
        numempty        /* int_n_sign_posn */
};

static struct lc_monetary_T _monetary_locale;
static int      _monetary_using_locale;
static char     *_monetary_locale_buf;

struct xlocale_monetary __xlocale_global_monetary;

#endif /* ! codereview */
static char
cnv(const char *str)
{
        int i = strtol(str, NULL, 10);

        if (i == -1)
                i = CHAR_MAX;
        return ((char)i);
}

static void
destruct_monetary(void *v)
{
        struct xlocale_monetary *l = v;

        if (l->buffer != NULL)
                free(l->buffer);

        free(l);
}

static int
monetary_load_locale_l(struct xlocale_monetary *loc, int *using_locale,
    int *changed, const char *name)
int
__monetary_load_locale(const char *name)
{
        int ret;
        struct lc_monetary_T *l = &loc->locale;
#endif /* ! codereview */

        ret = __part_load_locale(name, using_locale,
            &loc->buffer, "LC_MONETARY",
        ret = __part_load_locale(name, &_monetary_using_locale,
            &_monetary_locale_buf, "LC_MONETARY",
            LCMONETARY_SIZE_FULL, LCMONETARY_SIZE_MIN,
            (const char **)l);
            (const char **)&_monetary_locale);
        if (ret != _LDP_ERROR)
                *changed = 1;
                __mlocale_changed = 1;
        if (ret == _LDP_LOADED) {
                l->mon_grouping =
                    __fix_locale_grouping_str(l->mon_grouping);
```

```
 55                 _monetary_locale.mon_grouping =
 56                     __fix_locale_grouping_str(_monetary_locale.mon_grouping);

122 #define M_ASSIGN_CHAR(NAME) \
123                 (((char *)l->NAME)[0] = \
124                         cnv(l->NAME))
 59                 (((char *)_monetary_locale.NAME)[0] = \
 60                         cnv(_monetary_locale.NAME))

126                 M_ASSIGN_CHAR(int_frac_digits);
127                 M_ASSIGN_CHAR(frac_digits);
128                 M_ASSIGN_CHAR(p_cs_precedes);
129                 M_ASSIGN_CHAR(p_sep_by_space);
130                 M_ASSIGN_CHAR(n_cs_precedes);
131                 M_ASSIGN_CHAR(n_sep_by_space);
132                 M_ASSIGN_CHAR(p_sign_posn);
133                 M_ASSIGN_CHAR(n_sign_posn);

135                 /*
136                  * The six additional C99 international monetary formatting
137                  * parameters default to the national parameters when
138                  * reading FreeBSD LC_MONETARY data files.
139                  */
140 #define M_ASSIGN_ICHAR(NAME)                                    \
141                 do {                                            \
142                         if (l->int_##NAME == NULL)              \
143                                 l->int_##NAME =                 \
144                                         l->NAME;                \
 77                 if (_monetary_locale.int_##NAME == NULL)        \
 78                         _monetary_locale.int_##NAME =           \
 79                             _monetary_locale.NAME;              \
145                         else                                    \
146                                 M_ASSIGN_CHAR(int_##NAME);      \
147                 } while (0)
 81                         M_ASSIGN_CHAR(int_##NAME);

149                 M_ASSIGN_ICHAR(p_cs_precedes);
150                 M_ASSIGN_ICHAR(n_cs_precedes);
151                 M_ASSIGN_ICHAR(p_sep_by_space);
152                 M_ASSIGN_ICHAR(n_sep_by_space);
153                 M_ASSIGN_ICHAR(p_sign_posn);
154                 M_ASSIGN_ICHAR(n_sign_posn);
155         }
156         return (ret);
157 }

159 int
160 __monetary_load_locale(const char *name)
161 {
162         return (monetary_load_locale_l(&__xlocale_global_monetary,
163             &__xlocale_global_locale.using_monetary_locale,
164             &__xlocale_global_locale.monetary_locale_changed, name));
165 }

167 void *
168 __monetary_load(const char *name, locale_t loc)
169 {
170         struct xlocale_monetary *new;

172         new = calloc(sizeof(struct xlocale_monetary), 1);
173         if (new == NULL)
174                 return (NULL);

176         new->header.header.destructor = destruct_monetary;
177         if (monetary_load_locale_l(new, &loc->using_monetary_locale,
178             &loc->monetary_locale_changed, name) == _LDP_ERROR) {
```



```
179                 xlocale_release(new);
180                 return (NULL);
181         }

183         return (NULL);
184 }

186 #endif /* ! codereview */
187 struct lc_monetary_T *
188 __get_current_monetary_locale(locale_t loc)
 93 __get_current_monetary_locale(void)
189 {
190         return (loc->using_monetary_locale
191             ? &((struct xlocale_monetary*)loc->components[XLC_MONETARY])->locale
192             : (struct lc_monetary_T *)&_C_monetary_locale);
 95         return (_monetary_using_locale ? &_monetary_locale :
 96             (struct lc_monetary_T *)&_C_monetary_locale);
193 }
```

_____*unchanged_portion_omitted_*

```
**********************************************************
    2531 Thu Jul 11 12:26:12 2013
new/usr/src/lib/libc/port/locale/lmonetary.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _LMONETARY_H_
#define _LMONETARY_H_

#include "xlocale_private.h"

#endif /* ! codereview */
struct lc_monetary_T {
        const char      *int_curr_symbol;
        const char      *currency_symbol;
        const char      *mon_decimal_point;
        const char      *mon_thousands_sep;
        const char      *mon_grouping;
        const char      *positive_sign;
        const char      *negative_sign;
        const char      *int_frac_digits;
        const char      *frac_digits;
        const char      *p_cs_precedes;
        const char      *p_sep_by_space;
        const char      *n_cs_precedes;
        const char      *n_sep_by_space;
        const char      *p_sign_posn;
        const char      *n_sign_posn;
        const char      *int_p_cs_precedes;
        const char      *int_n_cs_precedes;
        const char      *int_p_sep_by_space;
        const char      *int_n_sep_by_space;
        const char      *int_p_sign_posn;
        const char      *int_n_sign_posn;
};
```



```
struct xlocale_monetary {
        struct xlocale_component header;
        char *buffer;
        struct lc_monetary_T locale;
};

struct lc_monetary_T *__get_current_monetary_locale(locale_t);
struct lc_monetary_T *__get_current_monetary_locale(void);
int     __monetary_load_locale(const char *);

#endif /* !_LMONETARY_H_ */
```

```
**********************************************************
    3881 Thu Jul 11 12:26:12 2013
new/usr/src/lib/libc/port/locale/lnumeric.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <limits.h>
#include "ldpart.h"
#include "lnumeric.h"
#include "../i18n/_locale.h"

extern int __nlocale_changed;
extern const char *__fix_locale_grouping_str(const char *);

#define LCNUMERIC_SIZE (sizeof (struct lc_numeric_T) / sizeof (char *))

static char     numempty[] = { CHAR_MAX, '\0' };

static const struct lc_numeric_T _C_numeric_locale = {
        ".",            /* decimal_point */
        "",             /* thousands_sep */
        numempty        /* grouping */
};

static void
destruct_numeric(void *v)
{
        struct xlocale_numeric *l = v;

        if (l->buffer != NULL)
                free(l->buffer);

        free(l);
```



```
}

struct xlocale_numeric __xlocale_global_numeric;
static struct lc_numeric_T _numeric_locale;
static int      _numeric_using_locale;
static char     *_numeric_locale_buf;

static int
numeric_load_locale(struct xlocale_numeric *loc, int *using_locale,
    int *changed, const char *name)
int
__numeric_load_locale(const char *name)
{
        int leg;
        const struct lc_numeric_T *leg = &_C_numeric_locale;

        int ret;
        struct lc_numeric_T *l = &loc->locale;
#endif /* ! codereview */

        ret = __part_load_locale(name, using_locale,
            &loc->buffer, "LC_NUMERIC",
            LCNUMERIC_SIZE, LCNUMERIC_SIZE,
            (const char **)l);
        if (ret != _LDP_ERROR)
                *changed = 1;
        ret = __part_load_locale(name, &_numeric_using_locale,
            &_numeric_locale_buf, "LC_NUMERIC", LCNUMERIC_SIZE, LCNUMERIC_SIZE,
            (const char **)&_numeric_locale);
        if (ret == _LDP_ERROR)
                return (_LDP_ERROR);

        __nlocale_changed = 1;
        if (ret == _LDP_LOADED) {
                /* Can't be empty according to C99 */
                if (*l->decimal_point == '\0')
                        l->decimal_point =
                if (*_numeric_locale.decimal_point == '\0')
                        _numeric_locale.decimal_point =
                            _C_numeric_locale.decimal_point;
                l->grouping =
                    __fix_locale_grouping_str(l->grouping);
                // XXX leg = (const struct lc_numeric_T *)&_numeric_locale;
                _numeric_locale.grouping =
                    __fix_locale_grouping_str(_numeric_locale.grouping);
                leg = (const struct lc_numeric_T *)&_numeric_locale;
        }
        /* This is Solaris legacy, required for ABI compatability */
        // _numeric[0] = *leg->decimal_point;
        // _numeric[1] = *leg->grouping;
        _numeric[0] = *leg->decimal_point;
        _numeric[1] = *leg->grouping;
        return (ret);
}

int
__numeric_load_locale(const char *name)
{
        return (numeric_load_locale(&__xlocale_global_numeric,
            &__xlocale_global_locale.using_numeric_locale,
            &__xlocale_global_locale.numeric_locale_changed, name));
}

void *
__numeric_load(const char *name, locale_t l)
{
```

```
107         struct xlocale_numeric *new;

109         new = calloc(sizeof(struct xlocale_numeric), 1);
110         if (new == NULL)
111                 return (NULL);

113         new->header.header.destructor = destruct_numeric;
114         if (numeric_load_locale(new, &l->using_numeric_locale,
115             &l->numeric_locale_changed, name) == _LDP_ERROR) {
116                 xlocale_release(new);
117                 return (NULL);
118         }

120         return (new);
121 }

123 #endif /* ! codereview */
124 struct lc_numeric_T *
125 __get_current_numeric_locale(locale_t loc)
 51 __get_current_numeric_locale(void)
126 {
127         return (loc->using_numeric_locale
128             ? &((struct xlocale_numeric *)loc->components[XLC_NUMERIC])->locale
129             : (struct lc_numeric_T *)&_C_numeric_locale);
 53         return (_numeric_using_locale ? &_numeric_locale :
 54             (struct lc_numeric_T *)&_C_numeric_locale);
130 }
```

_____*unchanged_portion_omitted_*

```
**********************************************************
    1982 Thu Jul 11 12:26:12 2013
new/usr/src/lib/libc/port/locale/lnumeric.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
 1 /*
 2  * Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
 3  * All rights reserved.
 4  *
 5  * Copyright (c) 2011 The FreeBSD Foundation
 6  * All rights reserved.
 7  * Portions of this software were developed by David Chisnall
 8  * under sponsorship from the FreeBSD Foundation.
 9  *
10 #endif /* ! codereview */
11  * Redistribution and use in source and binary forms, with or without
12  * modification, are permitted provided that the following conditions
13  * are met:
14  * 1. Redistributions of source code must retain the above copyright
15  *    notice, this list of conditions and the following disclaimer.
16  * 2. Redistributions in binary form must reproduce the above copyright
17  *    notice, this list of conditions and the following disclaimer in the
18  *    documentation and/or other materials provided with the distribution.
19  *
20  * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
21  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
22  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
23  * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
24  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
25  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
26  * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
27  * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
28  * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
29  * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
30  * SUCH DAMAGE.
31  */

33 #ifndef _LNUMERIC_H_
34 #define _LNUMERIC_H_

36 #include "xlocale_private.h"

38 #endif /* ! codereview */
39 struct lc_numeric_T {
40         const char      *decimal_point;
41         const char      *thousands_sep;
42         const char      *grouping;
43 };

45 struct xlocale_numeric {
46         struct xlocale_component header;
47         char *buffer;
48         struct lc_numeric_T locale;
49 };

51 struct lc_numeric_T *__get_current_numeric_locale(locale_t);
 5 struct lc_numeric_T *__get_current_numeric_locale(void);
52 int     __numeric_load_locale(const char *);

54 #endif /* !_LNUMERIC_H_ */
```

```
**********************************************************
    3967 Thu Jul 11 12:26:12 2013
new/usr/src/lib/libc/port/locale/localeconv.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * Copyright (c) 1991, 1993
 *      The Regents of the University of California.  All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 4. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _LCONV_C99
#define _LCONV_C99      /* so we get all the extensions */
#endif

#include "lint.h"
#include <locale.h>
#include "lmonetary.h"
#include "lnumeric.h"

/*
 * The localeconv() function constructs a struct lconv from the current
 * monetary and numeric locales.
 *
 * Because localeconv() may be called many times (especially by library
 * routines like printf() & strtod()), the approprate members of the
 * lconv structure are computed only when the monetary or numeric
 * locale has been changed.
 */
int __mlocale_changed = 1;
int __nlocale_changed = 1;

/*
 * Return the current locale conversion.
 */
```



```
struct lconv *
localeconv_l(locale_t loc)
localeconv(void)
{
        FIX_LOCALE(loc);
        struct lconv *ret = &loc->lconv;
        static struct lconv ret;

        if (loc->monetary_locale_changed) {
        if (__mlocale_changed) {
                /* LC_MONETARY part */
                struct lc_monetary_T *mptr;

#define M_ASSIGN_STR(NAME) (ret->NAME = (char *)mptr->NAME)
#define M_ASSIGN_CHAR(NAME) (ret->NAME = mptr->NAME[0])
#define M_ASSIGN_STR(NAME) (ret.NAME = (char *)mptr->NAME)
#define M_ASSIGN_CHAR(NAME) (ret.NAME = mptr->NAME[0])

                mptr = __get_current_monetary_locale(loc);
                mptr = __get_current_monetary_locale();
                M_ASSIGN_STR(int_curr_symbol);
                M_ASSIGN_STR(currency_symbol);
                M_ASSIGN_STR(mon_decimal_point);
                M_ASSIGN_STR(mon_thousands_sep);
                M_ASSIGN_STR(mon_grouping);
                M_ASSIGN_STR(positive_sign);
                M_ASSIGN_STR(negative_sign);
                M_ASSIGN_CHAR(int_frac_digits);
                M_ASSIGN_CHAR(frac_digits);
                M_ASSIGN_CHAR(p_cs_precedes);
                M_ASSIGN_CHAR(p_sep_by_space);
                M_ASSIGN_CHAR(n_cs_precedes);
                M_ASSIGN_CHAR(n_sep_by_space);
                M_ASSIGN_CHAR(p_sign_posn);
                M_ASSIGN_CHAR(n_sign_posn);
                M_ASSIGN_CHAR(int_p_cs_precedes);
                M_ASSIGN_CHAR(int_n_cs_precedes);
                M_ASSIGN_CHAR(int_p_sep_by_space);
                M_ASSIGN_CHAR(int_n_sep_by_space);
                M_ASSIGN_CHAR(int_p_sign_posn);
                M_ASSIGN_CHAR(int_n_sign_posn);
                loc->monetary_locale_changed = 0;
                __mlocale_changed = 0;
        }

        if (loc->numeric_locale_changed) {
        if (__nlocale_changed) {
                /* LC_NUMERIC part */
                struct lc_numeric_T *nptr;

#define N_ASSIGN_STR(NAME) (ret->NAME = (char *)nptr->NAME)
#define N_ASSIGN_STR(NAME) (ret.NAME = (char *)nptr->NAME)

                nptr = __get_current_numeric_locale(loc);
                nptr = __get_current_numeric_locale();
                N_ASSIGN_STR(decimal_point);
                N_ASSIGN_STR(thousands_sep);
                N_ASSIGN_STR(grouping);
                loc->numeric_locale_changed = 0;
                __nlocale_changed = 0;
        }

        return (ret);
}

struct lconv *
```

```
115 localeconv(void)
116 {
117         return (localeconv_l(__get_locale()));
 63         return (&ret);
118 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
    3859 Thu Jul 11 12:26:13 2013
new/usr/src/lib/libc/port/locale/mblocal.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2004 Tim J. Robbins.
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _MBLOCAL_H_
#define _MBLOCAL_H_

#include "runetype.h"
#include "xlocale_private.h"

/*
 * Conversion function pointers for current encoding.
 */
struct xlocale_ctype {
        struct xlocale_component header;
        _RuneLocale *runes;
        size_t (*__mbrtowc)(wchar_t *_RESTRICT_KYWD, const char *_RESTRICT_KYWD,
            size_t, mbstate_t *_RESTRICT_KYWD);
        int (*__mbsinit)(const mbstate_t *);
        size_t (*__mbsnrtowcs)(wchar_t *_RESTRICT_KYWD, const char **_RESTRICT_K
            size_t, size_t, mbstate_t *_RESTRICT_KYWD);
        size_t (*__wcrtomb)(char *_RESTRICT_KYWD, wchar_t, mbstate_t *_RESTRICT_
        size_t (*__wcsnrtombs)(char *_RESTRICT_KYWD, const wchar_t **_RESTRICT_K
            size_t, size_t, mbstate_t *_RESTRICT_KYWD);
        int __mb_cur_max;
        int __mb_sb_limit;
};
#define XLOCALE_CTYPE(x)        ((struct xlocale_ctype*)(x)->components[XLC_CTYP
extern struct xlocale_ctype __xlocale_global_ctype;
#endif /* ! codereview */

/*
```



```
 * Rune initialization function prototypes.
 */
int     _none_init(struct xlocale_ctype *, _RuneLocale *);
int     _none_init(_RuneLocale *);
int     _UTF8_init(_RuneLocale *);
int     _EUC_CN_init(_RuneLocale *);
int     _EUC_JP_init(_RuneLocale *);
int     _EUC_KR_init(_RuneLocale *);
int     _EUC_TW_init(_RuneLocale *);
int     _GB18030_init(_RuneLocale *);
int     _GB2312_init(_RuneLocale *);
int     _GBK_init(_RuneLocale *);
int     _BIG5_init(_RuneLocale *);
int     _MSKanji_init(_RuneLocale *);

/*
 * Conversion function pointers for current encoding.
 */
extern size_t (*__mbrtowc)(wchar_t *_RESTRICT_KYWD,
    const char *_RESTRICT_KYWD, size_t, mbstate_t *_RESTRICT_KYWD);
extern int (*__mbsinit)(const mbstate_t *);
extern size_t (*__mbsnrtowcs)(wchar_t *_RESTRICT_KYWD,
    const char **_RESTRICT_KYWD, size_t, size_t, mbstate_t *_RESTRICT_KYWD);

extern size_t (*__wcrtomb)(char *_RESTRICT_KYWD, wchar_t,
    mbstate_t *_RESTRICT_KYWD);

extern size_t (*__wcsnrtombs)(char *_RESTRICT_KYWD,
    const wchar_t **_RESTRICT_KYWD, size_t, size_t, mbstate_t *_RESTRICT_KYWD);

extern int charset_is_ascii;

size_t __mbsnrtowcs_std(wchar_t *_RESTRICT_KYWD, const char **_RESTRICT_KYWD,
    size_t, size_t, mbstate_t *_RESTRICT_KYWD);
size_t __wcsnrtombs_std(char *_RESTRICT_KYWD, const wchar_t **_RESTRICT_KYWD,
    size_t, size_t, mbstate_t *_RESTRICT_KYWD);

#define MIN(a, b)       ((a) < (b) ? (a) : (b))

#endif  /* _MBLOCAL_H_ */
```

```
**********************************************************
    5785 Thu Jul 11 12:26:13 2013
new/usr/src/lib/libc/port/locale/nl_langinfo.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2001, 2003 Alexey Zelkin <phantom@FreeBSD.org>
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <langinfo.h>
#include <limits.h>
#include <locale.h>
#include <stdlib.h>
#include <string.h>

#include "lnumeric.h"
#include "lmessages.h"
#include "lmonetary.h"
#include "timelocal.h"

#define _REL(BASE) ((int)item-BASE)

#define MONETARY        (__get_current_monetary_locale(loc))
#define TIME            (__get_current_time_locale(loc))
#define MESSAGES        (__get_current_messages_locale(loc))    /* XXX */
#define NUMERIC         (__get_current_numeric_locale(loc))
#define MONETARY        (__get_current_monetary_locale())
#define TIME            (__get_current_time_locale())
#define MESSAGES        (__get_current_messages_locale())
#define NUMERIC         (__get_current_numeric_locale())

#pragma weak _nl_langinfo = nl_langinfo

char *
nl_langinfo_l(nl_item item, locale_t loc)
nl_langinfo(nl_item item)
```



```
{
        char *ret, *s, *cs;
        static char *csym = NULL;

        switch (item) {
        case CODESET:
                ret = "";
                /*
                 * The codeset is the suffix of a locale, for most it will
                 * will be UTF-8, as in "en.UTF-8".  Short form locales are
                 * not supported.  Note also that although FreeBSD uses
                 * US-ASCII, Solaris historically has reported "646" for the
                 * C locale.
                 */
                if ((s = setlocale(LC_CTYPE, NULL)) != NULL) {
                        if ((cs = strchr(s, '.')) != NULL)
                                ret = cs + 1;
                        else if (strcmp(s, "C") == 0 || strcmp(s, "POSIX") == 0)
                                ret = "646";
                }
                break;
        case D_T_FMT:
                ret = (char *)TIME->c_fmt;
                break;
        case D_FMT:
                ret = (char *)TIME->x_fmt;
                break;
        case T_FMT:
                ret = (char *)TIME->X_fmt;
                break;
        case T_FMT_AMPM:
                ret = (char *)TIME->ampm_fmt;
                break;
        case AM_STR:
                ret = (char *)TIME->am;
                break;
        case PM_STR:
                ret = (char *)TIME->pm;
                break;
        case DAY_1: case DAY_2: case DAY_3:
        case DAY_4: case DAY_5: case DAY_6: case DAY_7:
                ret = (char *)TIME->weekday[_REL(DAY_1)];
                break;
        case ABDAY_1: case ABDAY_2: case ABDAY_3:
        case ABDAY_4: case ABDAY_5: case ABDAY_6: case ABDAY_7:
                ret = (char *)TIME->wday[_REL(ABDAY_1)];
                break;
        case MON_1: case MON_2: case MON_3: case MON_4:
        case MON_5: case MON_6: case MON_7: case MON_8:
        case MON_9: case MON_10: case MON_11: case MON_12:
                ret = (char *)TIME->month[_REL(MON_1)];
                break;
        case ABMON_1: case ABMON_2: case ABMON_3: case ABMON_4:
        case ABMON_5: case ABMON_6: case ABMON_7: case ABMON_8:
        case ABMON_9: case ABMON_10: case ABMON_11: case ABMON_12:
                ret = (char *)TIME->mon[_REL(ABMON_1)];
                break;
        case ERA:
                /* XXX: need to be implemented  */
                ret = "";
                break;
        case ERA_D_FMT:
                /* XXX: need to be implemented  */
                ret = "";
                break;
        case ERA_D_T_FMT:
```

```
123                 /* XXX: need to be implemented  */
124                 ret = "";
125                 break;
126         case ERA_T_FMT:
127                 /* XXX: need to be implemented  */
128                 ret = "";
129                 break;
130         case ALT_DIGITS:
131                 /* XXX: need to be implemented  */
132                 ret = "";
133                 break;
134         case RADIXCHAR:
135                 ret = (char *)NUMERIC->decimal_point;
136                 break;
137         case THOUSEP:
138                 ret = (char *)NUMERIC->thousands_sep;
139                 break;
140         case YESEXPR:
141                 ret = (char *)MESSAGES->yesexpr;
142                 break;
143         case NOEXPR:
144                 ret = (char *)MESSAGES->noexpr;
145                 break;
146         /*
147          * YESSTR and NOSTR items marked with LEGACY are available, but not
148          * recomended by SUSv2 to be used in portable applications since
149          * they're subject to remove in future specification editions.
150          */
151         case YESSTR:    /* LEGACY  */
152                 ret = (char *)MESSAGES->yesstr;
153                 break;
154         case NOSTR:     /* LEGACY  */
155                 ret = (char *)MESSAGES->nostr;
156                 break;
157         /*
158          * SUSv2 special formatted currency string
159          */
160         case CRNCYSTR:
161                 ret = "";
162                 cs = (char *)MONETARY->currency_symbol;
163                 if (*cs != '\0') {
164                         char pos = localeconv()->p_cs_precedes;

166                         if (pos == localeconv()->n_cs_precedes) {
167                                 char psn = '\0';

169                                 if (pos == CHAR_MAX) {
170                                         if (strcmp(cs,
171                                             MONETARY->mon_decimal_point) == 0)
172                                                 psn = '.';
173                                 } else
174                                         psn = pos ? '-' : '+';
175                                 if (psn != '\0') {
176                                         int clen = strlen(cs);
177                                         char *newc;

179                                         newc = realloc(csym, clen + 2);
180                                         if (newc != NULL) {
181                                                 free(csym);
182                                                 csym = newc;
183                                                 *csym = psn;
184                                                 (void) strcpy(csym + 1, cs);
185                                                 ret = csym;
186                                         }
187                                 }
188                         }
```



```
189                 }
190                 break;
191         case _DATE_FMT:         /* Solaris specific extension */
192                 ret = (char *)TIME->date_fmt;
193                 break;
194         /*
195          * Note that FreeBSD also had a private D_MD_ORDER, but that appears
196          * to have been specific to FreeBSD, so we have not included it here.
197          */
198         default:
199                 ret = "";
200         }
201         return (ret);
202 }

204 char *
205 nl_langinfo(nl_item item)
206 {
207         return (nl_langinfo_l(item, __get_locale()));
208 }
209 #endif /* ! codereview */
```

```
**********************************************************
    6220 Thu Jul 11 12:26:13 2013
new/usr/src/lib/libc/port/locale/none.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2002-2004 Tim J. Robbins. All rights reserved.
 * Copyright (c) 1993
 *      The Regents of the University of California.  All rights reserved.
 *
 * This code is derived from software contributed to Berkeley by
 * Paul Borman at Krystal Technologies.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 4. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <errno.h>
#include <limits.h>
#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <wchar.h>
#include <note.h>
#include "runetype.h"
#include "mblocal.h"

static size_t   _none_mbrtowc(wchar_t *_RESTRICT_KYWD,
    const char *_RESTRICT_KYWD, size_t, mbstate_t *_RESTRICT_KYWD);

static int      _none_mbsinit(const mbstate_t *);
static size_t   _none_mbsnrtowcs(wchar_t *_RESTRICT_KYWD dst,
    const char **_RESTRICT_KYWD src, size_t nms, size_t len,
    mbstate_t *_RESTRICT_KYWD);
static size_t   _none_wcrtomb(char *_RESTRICT_KYWD, wchar_t,
    mbstate_t *_RESTRICT_KYWD);
```



```
static size_t   _none_wcsnrtombs(char *_RESTRICT_KYWD,
    const wchar_t **_RESTRICT_KYWD,
    size_t, size_t, mbstate_t *_RESTRICT_KYWD);

/* setup defaults */

int
_none_init(struct xlocale_ctype *l, _RuneLocale *rl)
_none_init(_RuneLocale *rl)
{
        charset_is_ascii = 1;   /* XXX */
        charset_is_ascii = 1;

        l->__mbrtowc = _none_mbrtowc;
        l->__mbsinit = _none_mbsinit;
        l->__mbsnrtowcs = _none_mbsnrtowcs;
        l->__wcrtomb = _none_wcrtomb;
        l->__wcsnrtombs = _none_wcsnrtombs;
        l->runes = rl;
        /* XXX */
        __mbrtowc = _none_mbrtowc;
        __mbsinit = _none_mbsinit;
        __mbsnrtowcs = _none_mbsnrtowcs;
        __wcrtomb = _none_wcrtomb;
        __wcsnrtombs = _none_wcsnrtombs;
        _CurrentRuneLocale = rl;
        return (0);
}
_____unchanged_portion_omitted_

/* setup defaults */

size_t (*__mbrtowc)(wchar_t *_RESTRICT_KYWD, const char *_RESTRICT_KYWD,
    size_t, mbstate_t *_RESTRICT_KYWD) = _none_mbrtowc;

int (*__mbsinit)(const mbstate_t *) = _none_mbsinit;

size_t (*__mbsnrtowcs)(wchar_t *_RESTRICT_KYWD, const char **_RESTRICT_KYWD,
    size_t, size_t, mbstate_t *_RESTRICT_KYWD) = _none_mbsnrtowcs;

size_t (*__wcrtomb)(char *_RESTRICT_KYWD, wchar_t, mbstate_t *_RESTRICT_KYWD) =
    _none_wcrtomb;

size_t (*__wcsnrtombs)(char *_RESTRICT_KYWD, const wchar_t **_RESTRICT_KYWD,
    size_t, size_t, mbstate_t *_RESTRICT_KYWD) = _none_wcsnrtombs;

struct xlocale_ctype __xlocale_global_ctype = {
        {{0}, "C"},
        (_RuneLocale*)&_DefaultRuneLocale,
        _none_mbrtowc,
        _none_mbsinit,
        _none_mbsnrtowcs,
        _none_wcrtomb,
        _none_wcsnrtombs,
        1,      /* __mb_cur_max */
        256     /* __mb_sb_limit */
};

const struct xlocale_ctype __xlocale_C_ctype = {
        {{0}, "C"},
        (_RuneLocale*)&_DefaultRuneLocale,
        _none_mbrtowc,
        _none_mbsinit,
        _none_mbsnrtowcs,
        _none_wcrtomb,
        _none_wcsnrtombs,
```

```
228         1,      /* __mb_cur_max */
229         256     /* __mb_sb_limit */
230 };
231 #endif /* ! codereview */
```

```
**********************************************************
    8205 Thu Jul 11 12:26:13 2013
new/usr/src/lib/libc/port/locale/setlocale.c
2964 need POSIX 2008 locale object support
**********************************************************
______unchanged_portion_omitted_

  83 /*
  84  * Path to locale storage directory.  See ../i18n/_loc_path.h
  85  */
  86 char    *_PathLocale = _DFLT_LOC_PATH;

  88 /*
  89  * The locales we are going to try and load
  90  */
  91 static char new_categories[NUM_CATS][ENCODING_LEN + 1];
  92 static char saved_categories[NUM_CATS][ENCODING_LEN + 1];
  93 static char current_locale_string[NUM_CATS * (ENCODING_LEN + 1 + 1)];

  95 static char     *currentlocale(void);
  96 static char     *loadlocale(int);
  97 static const char *__get_locale_env(int);

  98 char *
  99 setlocale(int category, const char *locale)
 100 {
 101         int i, j, saverr;
 102         const char *env, *r;

 104         if (category < 0 || category >= NUM_CATS) {
 105                 errno = EINVAL;
 106                 return (NULL);
 107         }

 109         if (locale == NULL)
 110                 return (category != LC_ALL ?
 111                     current_categories[category] : currentlocale());

 113         /*
 114          * Default to the current locale for everything.
 115          */
 116         for (i = 0; i < NUM_CATS; ++i)
 117                 (void) strcpy(new_categories[i], current_categories[i]);

 119         /*
 120          * Now go fill up new_categories from the locale argument
 121          */
 122         if (!*locale) {
 123                 if (category == LC_ALL) {
 124                         for (i = 0; i < NUM_CATS; ++i) {
 125                                 if (i == LC_ALL)
 126                                         continue;
 127                                 env = __get_locale_env(i);
 128                                 if (strlen(env) > ENCODING_LEN) {
 129                                         errno = EINVAL;
 130                                         return (NULL);
 131                                 }
 132                                 (void) strcpy(new_categories[i], env);
 133                         }
 134                 } else {
 135                         env = __get_locale_env(category);
 136                         if (strlen(env) > ENCODING_LEN) {
 137                                 errno = EINVAL;
 138                                 return (NULL);
 139                         }
 140                         (void) strcpy(new_categories[category], env);
```



```
 141                 }
 142         } else if (category != LC_ALL) {
 143                 if (strlen(locale) > ENCODING_LEN) {
 144                         errno = EINVAL;
 145                         return (NULL);
 146                 }
 147                 (void) strcpy(new_categories[category], locale);
 148         } else {
 149                 if ((r = strchr(locale, '/')) == NULL) {
 150                         if (strlen(locale) > ENCODING_LEN) {
 151                                 errno = EINVAL;
 152                                 return (NULL);
 153                         }
 154                         for (i = 0; i < NUM_CATS; ++i)
 155                                 (void) strcpy(new_categories[i], locale);
 156                 } else {
 157                         char    *buf;
 158                         char    *save;

 160                         buf = alloca(strlen(locale) + 1);
 161                         (void) strcpy(buf, locale);

 163                         save = NULL;
 164                         r = strtok_r(buf, "/", &save);
 165                         for (i = 0;  i < NUM_CATS; i++) {
 166                                 if (i == LC_ALL)
 167                                         continue;
 168                                 if (r == NULL) {
 169                                         /*
 170                                          * Composite Locale is inadequately
 171                                          * specified!   (Or with empty fields.)
 172                                          * The old code would fill fields
 173                                          * out from the last one, but I think
 174                                          * this is suboptimal.
 175                                          */
 176                                         errno = EINVAL;
 177                                         return (NULL);
 178                                 }
 179                                 (void) strlcpy(new_categories[i], r,
 180                                     ENCODING_LEN);
 181                                 r = strtok_r(NULL, "/", &save);
 182                         }
 183                         if (r != NULL) {
 184                                 /*
 185                                  * Too many components - we had left over
 186                                  * data in the LC_ALL.  It is malformed.
 187                                  */
 188                                 errno = EINVAL;
 189                                 return (NULL);
 190                         }
 191                 }
 192         }

 194         if (category != LC_ALL)
 195                 return (loadlocale(category));

 197         for (i = 0; i < NUM_CATS; ++i) {
 198                 (void) strcpy(saved_categories[i], current_categories[i]);
 199                 if (i == LC_ALL)
 200                         continue;
 201                 if (loadlocale(i) == NULL) {
 202                         saverr = errno;
 203                         for (j = 0; j < i; j++) {
 204                                 (void) strcpy(new_categories[j],
 205                                     saved_categories[j]);
 206                                 if (i == LC_ALL)
```

```
207                                     continue;
208                             if (loadlocale(j) == NULL) {
209                                     (void) strcpy(new_categories[j], "C");
210                                     (void) loadlocale(j);
211                             }
212                     }
213                     errno = saverr;
214                     return (NULL);
215             }
216         }
217         return (currentlocale());
218 }
```

*unchanged_portion_omitted_*

```
307 const char *
308 static const char *
309 __get_locale_env(int category)
310 {
311         const char *env;

312         /* 1. check LC_ALL. */
313         env = getenv(categories[LC_ALL]);

314         /* 2. check LC_* */
315         if (env == NULL || !*env)
316                 env = getenv(categories[category]);

317         /* 3. check LANG */
318         if (env == NULL || !*env)
319                 env = getenv("LANG");

320         /* 4. if none is set, fall to "C" */
321         if (env == NULL || !*env)
322                 env = "C";

323         return (env);
324 }

325 /*
326  * Detect locale storage location and store its value to _PathLocale variable
327  */
328 int
329 __detect_path_locale(void)
330 {
331         /* XXX */

332         return (0);
333 }
334 #endif /* ! codereview */
```

```
**********************************************************
    1661 Thu Jul 11 12:26:13 2013
new/usr/src/lib/libc/port/locale/setlocale.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright (C) 1997 by Andrey A. Chernov, Moscow, Russia.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _SETLOCALE_H_
#define _SETLOCALE_H_

#define _PATH_LOCALE    "/usr/share/locale"
#define ENCODING_LEN 31
#define CATEGORY_LEN 11

extern char *_PathLocale;

const char *__get_locale_env(int);
#endif /* ! codereview */
int     __detect_path_locale(void);
int     __wrap_setrunelocale(const char *);

#endif /* !_SETLOCALE_H_ */
```

```
**********************************************************
    9099 Thu Jul 11 12:26:14 2013
new/usr/src/lib/libc/port/locale/setrunelocale.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1993
 *      The Regents of the University of California.  All rights reserved.
 *
 * This code is derived from software contributed to Berkeley by
 * Paul Borman at Krystal Technologies.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 4. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include "file64.h"
#include <errno.h>
#include <limits.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <wchar.h>
#include "runetype.h"
#include "ldpart.h"
#include "mblocal.h"
#include "setlocale.h"
#include "_ctype.h"
#include "../i18n/_locale.h"

/*
 * A cached version of the runes for this thread.  Used by ctype.h
 */
__thread const _RuneLocale *_ThreadRuneLocale;

#endif /* ! codereview */
```



```
extern _RuneLocale      *_Read_RuneMagi(FILE *);
extern unsigned char    __ctype_C[];

static int              __setrunelocale(struct xlocale_ctype *, const char *);
static int              __setrunelocale(const char *);

static int
__setrunelocale(struct xlocale_ctype *l, const char *encoding)
__setrunelocale(const char *encoding)
{
        FILE *fp;
        char name[PATH_MAX];
        _RuneLocale *rl;
        int saverr, ret;
        struct xlocale_ctype saved = *l; /* XXX DOUBLE NOT USED */
#endif /* ! codereview */
        size_t (*old__mbrtowc)(wchar_t *_RESTRICT_KYWD,
            const char *_RESTRICT_KYWD, size_t, mbstate_t *_RESTRICT_KYWD);
        size_t (*old__wcrtomb)(char *_RESTRICT_KYWD, wchar_t,
            mbstate_t *_RESTRICT_KYWD);
        int (*old__mbsinit)(const mbstate_t *);
        size_t (*old__mbsnrtowcs)(wchar_t *_RESTRICT_KYWD,
            const char **_RESTRICT_KYWD, size_t, size_t,
            mbstate_t *_RESTRICT_KYWD);
        size_t (*old__wcsnrtombs)(char *_RESTRICT_KYWD,
            const wchar_t **_RESTRICT_KYWD, size_t, size_t,
            mbstate_t *_RESTRICT_KYWD);
        static char ctype_encoding[ENCODING_LEN + 1];
        static _RuneLocale *CachedRuneLocale;
        static size_t (*Cached__mbrtowc)(wchar_t *_RESTRICT_KYWD,
            const char *_RESTRICT_KYWD, size_t, mbstate_t *_RESTRICT_KYWD);
        static size_t (*Cached__wcrtomb)(char *_RESTRICT_KYWD, wchar_t,
            mbstate_t *_RESTRICT_KYWD);
        static int (*Cached__mbsinit)(const mbstate_t *);
        static size_t (*Cached__mbsnrtowcs)(wchar_t *_RESTRICT_KYWD,
            const char **_RESTRICT_KYWD, size_t, size_t,
            mbstate_t *_RESTRICT_KYWD);
        static size_t (*Cached__wcsnrtombs)(char *_RESTRICT_KYWD,
            const wchar_t **_RESTRICT_KYWD, size_t, size_t,
            mbstate_t *_RESTRICT_KYWD);

        /*
         * The "C" and "POSIX" locale are always here.
         */
        if (strcmp(encoding, "C") == 0 || strcmp(encoding, "POSIX") == 0) {
                int i;

                (void) memcpy(__ctype, __ctype_C, SZ_TOTAL);

                for (i = 0; i < _CACHED_RUNES; i++) {
                        __ctype_mask[i] = _DefaultRuneLocale.__runetype[i];
                        __trans_upper[i] = _DefaultRuneLocale.__mapupper[i];
                        __trans_lower[i] = _DefaultRuneLocale.__maplower[i];
                }

                (void) _none_init(l, &_DefaultRuneLocale);
                (void) _none_init(&_DefaultRuneLocale);
                return (0);
        }

        /*
         * If the locale name is the same as our cache, use the cache.
         */
        if (CachedRuneLocale != NULL &&
            strcmp(encoding, ctype_encoding) == 0) {
                l->runes = CachedRuneLocale;
```

```
125                 l->__mbrtowc = Cached__mbrtowc;
126                 l->__mbsinit = Cached__mbsinit;
127                 l->__mbsnrtowcs = Cached__mbsnrtowcs;
128                 l->__wcrtomb = Cached__wcrtomb;
129                 l->__wcsnrtombs = Cached__wcsnrtombs;
 27                 _CurrentRuneLocale = CachedRuneLocale;
 28                 __mbrtowc = Cached__mbrtowc;
 29                 __mbsinit = Cached__mbsinit;
 30                 __mbsnrtowcs = Cached__mbsnrtowcs;
 31                 __wcrtomb = Cached__wcrtomb;
 32                 __wcsnrtombs = Cached__wcsnrtombs;
130                 return (0);
131         }

133         /*
134          * Slurp the locale file into the cache.
135          */

137         (void) snprintf(name, sizeof (name), "%s/%s/LC_CTYPE/LCL_DATA",
138             _PathLocale, encoding);

140         if ((fp = fopen(name, "r")) == NULL)
141                 return (errno == 0 ? ENOENT : errno);

143         if ((rl = _Read_RuneMagi(fp)) == NULL) {
144                 saverr = (errno == 0 ? EINVAL : errno);
145                 (void) fclose(fp);
146                 return (saverr);
147         }
148         (void) fclose(fp);

150         old__mbrtowc = __mbrtowc;
151         old__mbsinit = __mbsinit;
152         old__mbsnrtowcs = __mbsnrtowcs;
153         old__wcrtomb = __wcrtomb;
154         old__wcsnrtombs = __wcsnrtombs;

156         l->__mbrtowc = NULL;
157         l->__mbsinit = NULL;
158         l->__mbsnrtowcs = __mbsnrtowcs_std;
159         l->__wcrtomb = NULL;
160         l->__wcsnrtombs = __wcsnrtombs_std;
 59         __mbrtowc = NULL;
 60         __mbsinit = NULL;
 61         __mbsnrtowcs = __mbsnrtowcs_std;
 62         __wcrtomb = NULL;
 63         __wcsnrtombs = __wcsnrtombs_std;

162         if (strcmp(rl->__encoding, "NONE") == 0)
163                 ret = _none_init(l, rl);
 66                 ret = _none_init(rl);
164         else if (strcmp(rl->__encoding, "UTF-8") == 0)
165                 ret = _UTF8_init(rl);
166         else if (strcmp(rl->__encoding, "EUC-CN") == 0)
167                 ret = _EUC_CN_init(rl);
168         else if (strcmp(rl->__encoding, "EUC-JP") == 0)
169                 ret = _EUC_JP_init(rl);
170         else if (strcmp(rl->__encoding, "EUC-KR") == 0)
171                 ret = _EUC_KR_init(rl);
172         else if (strcmp(rl->__encoding, "EUC-TW") == 0)
173                 ret = _EUC_TW_init(rl);
174         else if (strcmp(rl->__encoding, "GB18030") == 0)
175                 ret = _GB18030_init(rl);
176         else if (strcmp(rl->__encoding, "GB2312") == 0)
177                 ret = _GB2312_init(rl);
178         else if (strcmp(rl->__encoding, "GBK") == 0)
```



```
179                 ret = _GBK_init(rl);
180         else if (strcmp(rl->__encoding, "BIG5") == 0)
181                 ret = _BIG5_init(rl);
182         else if (strcmp(rl->__encoding, "MSKanji") == 0)
183                 ret = _MSKanji_init(rl);
184         else
185                 ret = EINVAL;

187         if (ret == 0) {
188                 if (CachedRuneLocale != NULL) {
189                         free(CachedRuneLocale);
190                 }
191                 CachedRuneLocale = l->runes;
192                 Cached__mbrtowc = l->__mbrtowc;
193                 Cached__mbsinit = l->__mbsinit;
194                 Cached__mbsnrtowcs = l->__mbsnrtowcs;
195                 Cached__wcrtomb = l->__wcrtomb;
196                 Cached__wcsnrtombs = l->__wcsnrtombs;
 94                 CachedRuneLocale = _CurrentRuneLocale;
 95                 Cached__mbrtowc = __mbrtowc;
 96                 Cached__mbsinit = __mbsinit;
 97                 Cached__mbsnrtowcs = __mbsnrtowcs;
 98                 Cached__wcrtomb = __wcrtomb;
 99                 Cached__wcsnrtombs = __wcsnrtombs;
197                 (void) strcpy(ctype_encoding, encoding);

199                 /*
200                  * We need to overwrite the _ctype array.  This requires
201                  * some finagling.  This is because references to it may
202                  * have been baked into applications.
203                  *
204                  * Note that it is interesting that toupper/tolower only
205                  * produce defined results when the input is representable
206                  * as a byte.
207                  */

209                 /*
210                  * The top half is the type mask array.  Because we
211                  * want to support both legacy Solaris code (which have
212                  * mask valeus baked in to them), and we want to be able
213                  * to import locale files from other sources (FreeBSD)
214                  * which probably uses different masks, we have to perform
215                  * a conversion here.  Ugh.  Note that the _CTYPE definitions
216                  * we use from FreeBSD are richer than the Solaris legacy.
217                  *
218                  * We have to cope with these limitations though, because the
219                  * inadequate Solaris definitions were baked into binaries.
220                  */
221                 for (int i = 0; i < _CACHED_RUNES; i++) {
222                         /* ctype can only encode the lower 8 bits. */
223                         __ctype[i+1] = rl->__runetype[i] & 0xff;
224                         __ctype_mask[i] = rl->__runetype[i];
225                 }

227                 /* The bottom half is the toupper/lower array */
228                 for (int i = 0; i < _CACHED_RUNES; i++) {
229                         __ctype[258 + i] = i;
230                         if (rl->__mapupper[i] && rl->__mapupper[i] != i)
231                                 __ctype[258+i] = rl->__mapupper[i];
232                         if (rl->__maplower[i] && rl->__maplower[i] != i)
233                                 __ctype[258+i] = rl->__maplower[i];

235                         /* Don't forget these annoyances either! */
236                         __trans_upper[i] = rl->__mapupper[i];
237                         __trans_lower[i] = rl->__maplower[i];
238                 }
```

```
240                 /*
241                  * Note that we expect the init code will have populated
242                  * the CSWIDTH array (__ctype[514-520]) properly.
243                  */
244         } else {
245                 l->__mbrtowc = old__mbrtowc;
246                 l->__mbsinit = old__mbsinit;
247                 l->__mbsnrtowcs = old__mbsnrtowcs;
248                 l->__wcrtomb = old__wcrtomb;
249                 l->__wcsnrtombs = old__wcsnrtombs;
148                 __mbrtowc = old__mbrtowc;
149                 __mbsinit = old__mbsinit;
150                 __mbsnrtowcs = old__mbsnrtowcs;
151                 __wcrtomb = old__wcrtomb;
152                 __wcsnrtombs = old__wcsnrtombs;
250                 free(rl);
251         }

253         return (ret);
254 }

256 int
257 __wrap_setrunelocale(const char *locale)
258 {
259         int ret = __setrunelocale(&__xlocale_global_ctype, locale);
162         int ret = __setrunelocale(locale);

261         if (ret != 0) {
262                 errno = ret;
263                 return (_LDP_ERROR);
264         }
265         /* XXX */
266 //      __mb_cur_max = __xlocale_global_ctype.__mb_cur_max;
267 //      __mb_sb_limit = __xlocale_global_ctype.__mb_sb_limit;
268         _CurrentRuneLocale = __xlocale_global_ctype.runes;
269 #endif /* ! codereview */
270         return (_LDP_LOADED);
271 }

273 void
274 __set_thread_rune_locale(locale_t loc)
275 {

277         if (loc == NULL) {
278                 _ThreadRuneLocale = &_DefaultRuneLocale;
279         } else {
280                 _ThreadRuneLocale = XLOCALE_CTYPE(loc)->runes;
281         }
282 }

284 void *
285 __ctype_load(const char *locale, locale_t unused)
286 {
287         struct xlocale_ctype *l;

289         l = calloc(sizeof(struct xlocale_ctype), 1);
290         /* XXX */

292         return (l);
293 }
294 #endif /* ! codereview */
```

```
**********************************************************
   14282 Thu Jul 11 12:26:14 2013
new/usr/src/lib/libc/port/locale/strftime.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1989 The Regents of the University of California.
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms are permitted
 * provided that the above copyright notice and this paragraph are
 * duplicated in all such forms and that any documentation,
 * advertising materials, and other materials related to such
 * distribution and use acknowledge that the software was developed
 * by the University of California, Berkeley. The name of the
 * University may not be used to endorse or promote products derived
 * from this software without specific prior written permission.
 * THIS SOFTWARE IS PROVIDED ''AS IS'' AND WITHOUT ANY EXPRESS OR
 * IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
 */

#include "lint.h"
#include "tzfile.h"
#include <fcntl.h>
#include <sys/stat.h>
#include <string.h>
#include <stdio.h>
#include "timelocal.h"

static char *_add(const char *, char *, const char *);
static char *_conv(int, const char *, char *, const char *);
static char *_fmt(const char *, const struct tm *, char *, const char * const,
    int *, locale_t);
static char *_fmt(const char *, const struct tm *, char *, const char * const);
static char *_yconv(int, int, int, int, char *, const char *);

extern char *tzname[];

#define IN_NONE 0
#define IN_SOME 1
#define IN_THIS 2
#define IN_ALL  3

#define PAD_DEFAULT     0
#define PAD_LESS        1
#define PAD_SPACE       2
#define PAD_ZERO        3

static const char *fmt_padding[][4] = {
        /* DEFAULT,     LESS,   SPACE,  ZERO */
#define PAD_FMT_MONTHDAY        0
#define PAD_FMT_HMS             0
#define PAD_FMT_CENTURY         0
#define PAD_FMT_SHORTYEAR       0
#define PAD_FMT_MONTH           0
#define PAD_FMT_WEEKOFYEAR      0
#define PAD_FMT_DAYOFMONTH      0
        { "%02d",       "%d",   "%2d",  "%02d" },
```



```
#define PAD_FMT_SDAYOFMONTH     1
#define PAD_FMT_SHMS            1
        { "%2d",        "%d",   "%2d",  "%02d" },
#define PAD_FMT_DAYOFYEAR       2
        { "%03d",       "%d",   "%3d",  "%03d" },
#define PAD_FMT_YEAR            3
        { "%04d",       "%d",   "%4d",  "%04d" }
};


size_t
strftime_l(char *_RESTRICT_KYWD s, size_t maxsize,
    const char *_RESTRICT_KYWD format, const struct tm *_RESTRICT_KYWD t,
    locale_t loc)
strftime(char *_RESTRICT_KYWD s, size_t maxsize,
    const char *_RESTRICT_KYWD format, const struct tm *_RESTRICT_KYWD t)
{
        char    *p;
        int     warn;
        FIX_LOCALE(loc);
#endif /* ! codereview */

        tzset();
        warn = IN_NONE;
        p = _fmt(((format == NULL) ? "%c" : format), t, s, s + maxsize,
            &warn, loc);
        /* XXX YEAR_2000 */
        p = _fmt(((format == NULL) ? "%c" : format), t, s, s + maxsize);
        if (p == s + maxsize)
                return (0);
        *p = '\0';
        return (p - s);
}

size_t
strftime(char *_RESTRICT_KYWD s, size_t maxsize,
    const char *_RESTRICT_KYWD format, const struct tm *_RESTRICT_KYWD t)
{
        return (strftime_l(s, maxsize, format, t, __get_locale()));
}

#endif /* ! codereview */
static char *
_fmt(const char *format, const struct tm *t, char *pt, const char * const ptlim,
    int *warnp, locale_t loc)
_fmt(const char *format, const struct tm *t, char *pt, const char * const ptlim)
{
        int Ealternative, Oalternative, PadIndex;
        struct lc_time_T *tptr = __get_current_time_locale(loc);
        struct lc_time_T *tptr = __get_current_time_locale();

#define PADDING(x)      fmt_padding[x][PadIndex]

        for (; *format; ++format) {
                if (*format == '%') {
                        Ealternative = 0;
                        Oalternative = 0;
                        PadIndex         = PAD_DEFAULT;
label:
                        switch (*++format) {
                        case '\0':
                                --format;
                                break;
                        case 'A':
                                pt = _add((t->tm_wday < 0 ||
                                    t->tm_wday >= DAYSPERWEEK) ?
```

```
122                                 "?" : tptr->weekday[t->tm_wday],
123                                 pt, ptlim);
124                         continue;
125                 case 'a':
126                         pt = _add((t->tm_wday < 0 ||
127                             t->tm_wday >= DAYSPERWEEK) ?
128                             "?" : tptr->wday[t->tm_wday],
129                             pt, ptlim);
130                         continue;
131                 case 'B':
132                         pt = _add((t->tm_mon < 0 ||
133                             t->tm_mon >= MONSPERYEAR) ?
134                             "?" : (tptr->month)[t->tm_mon],
135                             pt, ptlim);
136                         continue;
137                 case 'b':
138                 case 'h':
139                         pt = _add((t->tm_mon < 0 ||
140                             t->tm_mon >= MONSPERYEAR) ?
141                             "?" : tptr->mon[t->tm_mon],
142                             pt, ptlim);
143                         continue;
144                 case 'C':
145                         /*
146                          * %C used to do a...
147                          *      _fmt("%a %b %e %X %Y", t);
148                          * ...whereas now POSIX 1003.2 calls for
149                          * something completely different.
150                          * (ado, 1993-05-24)
151                          */
152                         pt = _yconv(t->tm_year, TM_YEAR_BASE, 1, 0,
153                             pt, ptlim);
154                         continue;
155                 case 'c':
156                 {
157                         int warn2 = IN_SOME;

159                         pt = _fmt(tptr->c_fmt, t, pt, ptlim, &warn2,
160                             loc);
161                         /* XXX */
162                 }
107                         pt = _fmt(tptr->c_fmt, t, pt, ptlim);
163                         continue;
164                 case 'D':
165                         pt = _fmt("%m/%d/%y", t, pt, ptlim, warnp,
166                             loc);
110                         pt = _fmt("%m/%d/%y", t, pt, ptlim);
167                         continue;
168                 case 'd':
169                         pt = _conv(t->tm_mday,
170                             PADDING(PAD_FMT_DAYOFMONTH), pt, ptlim);
171                         continue;
172                 case 'E':
173                         if (Ealternative || Oalternative)
174                                 break;
175                         Ealternative++;
176                         goto label;
177                 case 'O':
178                         /*
179                          * C99 locale modifiers.
180                          * The sequences
181                          *      %Ec %EC %Ex %EX %Ey %EY
182                          *      %Od %oe %OH %OI %Om %OM
183                          *      %OS %Ou %OU %OV %Ow %OW %Oy
184                          * are supposed to provide alternate
185                          * representations.
```



```
186                          */
187                         if (Ealternative || Oalternative)
188                                 break;
189                         Oalternative++;
190                         goto label;
191                 case 'e':
192                         pt = _conv(t->tm_mday,
193                             PADDING(PAD_FMT_SDAYOFMONTH), pt, ptlim);
194                         continue;
195                 case 'F':
196                         pt = _fmt("%Y-%m-%d", t, pt, ptlim, warnp,
197                             loc);
140                         pt = _fmt("%Y-%m-%d", t, pt, ptlim);
198                         continue;
199                 case 'H':
200                         pt = _conv(t->tm_hour, PADDING(PAD_FMT_HMS),
201                             pt, ptlim);
202                         continue;
203                 case 'I':
204                         pt = _conv((t->tm_hour % 12) ?
205                             (t->tm_hour % 12) : 12,
206                             PADDING(PAD_FMT_HMS), pt, ptlim);
207                         continue;
208                 case 'j':
209                         pt = _conv(t->tm_yday + 1,
210                             PADDING(PAD_FMT_DAYOFYEAR), pt, ptlim);
211                         continue;
212                 case 'k':
213                         /*
214                          * This used to be...
215                          *      _conv(t->tm_hour % 12 ?
216                          *              t->tm_hour % 12 : 12, 2, ' ');
217                          * ...and has been changed to the below to
218                          * match SunOS 4.1.1 and Arnold Robbins'
219                          * strftime version 3.0. That is, "%k" and
220                          * "%l" have been swapped.
221                          * (ado, 1993-05-24)
222                          */
223                         pt = _conv(t->tm_hour,
224                             PADDING(PAD_FMT_SHMS), pt, ptlim);
225                         continue;
226                 case 'l':
227                         /*
228                          * This used to be...
229                          *      _conv(t->tm_hour, 2, ' ');
230                          * ...and has been changed to the below to
231                          * match SunOS 4.1.1 and Arnold Robbin's
232                          * strftime version 3.0. That is, "%k" and
233                          * "%l" have been swapped.
234                          * (ado, 1993-05-24)
235                          */
236                         pt = _conv((t->tm_hour % 12) ?
237                             (t->tm_hour % 12) : 12,
238                             PADDING(PAD_FMT_SHMS), pt, ptlim);
239                         continue;
240                 case 'M':
241                         pt = _conv(t->tm_min, PADDING(PAD_FMT_HMS),
242                             pt, ptlim);
243                         continue;
244                 case 'm':
245                         pt = _conv(t->tm_mon + 1,
246                             PADDING(PAD_FMT_MONTH),
247                             pt, ptlim);
248                         continue;
249                 case 'n':
250                         pt = _add("\n", pt, ptlim);
```

```
251                             continue;
252                     case 'p':
253                             pt = _add((t->tm_hour >= (HOURSPERDAY / 2)) ?
254                                 tptr->pm : tptr->am, pt, ptlim);
255                             continue;
256                     case 'R':
257                             pt = _fmt("%H:%M", t, pt, ptlim, warnp, loc);
200                             pt = _fmt("%H:%M", t, pt, ptlim);
258                             continue;
259                     case 'r':
260                             pt = _fmt(tptr->ampm_fmt, t, pt, ptlim,
261                                 warnp, loc);
203                             pt = _fmt(tptr->ampm_fmt, t, pt, ptlim);
262                             continue;
263                     case 'S':
264                             pt = _conv(t->tm_sec, PADDING(PAD_FMT_HMS),
265                                 pt, ptlim);
266                             continue;

268                     case 's':
269                             {
270                             struct tm tm;
271                             char *buf;
272 /* XXX */

273                             tm = *t;
274                             (void) asprintf(&buf, "%ld", mktime(&tm));
275                             pt = _add(buf, pt, ptlim);
276                             }
277 #endif /* ! codereview */
278                             continue;
218                     }

279                     case 'T':
280                             pt = _fmt("%H:%M:%S", t, pt, ptlim,
281                                 warnp, loc);
221                             pt = _fmt("%H:%M:%S", t, pt, ptlim);
282                             continue;
283                     case 't':
284                             pt = _add("\t", pt, ptlim);
285                             continue;
286                     case 'U':
287                             pt = _conv((t->tm_yday + DAYSPERWEEK -
288                                 t->tm_wday) / DAYSPERWEEK,
289                                 PADDING(PAD_FMT_WEEKOFYEAR),
290                                 pt, ptlim);
291                             continue;
292                     case 'u':
293                             /*
294                              * From Arnold Robbins' strftime version 3.0:
295                              * "ISO 8601: Weekday as a decimal number
296                              * [1 (Monday) - 7]"
297                              * (ado, 1993-05-24)
298                              */
299                             pt = _conv((t->tm_wday == 0) ?
300                                 DAYSPERWEEK : t->tm_wday,
301                                 "%d", pt, ptlim);
302                             continue;
303                     case 'V':       /* ISO 8601 week number */
304                     case 'G':       /* ISO 8601 year (four digits) */
305                     case 'g':       /* ISO 8601 year (two digits) */
306 /*
307  * From Arnold Robbins' strftime version 3.0: "the week number of the
308  * year (the first Monday as the first day of week 1) as a decimal number
309  * (01-53)."
310  * (ado, 1993-05-24)
```



```
311  *
312  * From "http://www.ft.uni-erlangen.de/~mskuhn/iso-time.html" by Markus Kuhn:
313  * "Week 01 of a year is per definition the first week which has the
314  * Thursday in this year, which is equivalent to the week which contains
315  * the fourth day of January. In other words, the first week of a new year
316  * is the week which has the majority of its days in the new year. Week 01
317  * might also contain days from the previous year and the week before week
318  * 01 of a year is the last week (52 or 53) of the previous year even if
319  * it contains days from the new year. A week starts with Monday (day 1)
320  * and ends with Sunday (day 7). For example, the first week of the year
321  * 1997 lasts from 1996-12-30 to 1997-01-05..."
322  * (ado, 1996-01-02)
323  */
324                         {
325                                 int     year;
326                                 int     base;
327                                 int     yday;
328                                 int     wday;
329                                 int     w;

331                                 year = t->tm_year;
332                                 base = TM_YEAR_BASE;
333                                 yday = t->tm_yday;
334                                 wday = t->tm_wday;
335                                 for (;;) {
336                                         int     len;
337                                         int     bot;
338                                         int     top;

340                                         len = isleap_sum(year, base) ?
341                                             DAYSPERLYEAR : DAYSPERNYEAR;
342                                         /*
343                                          * What yday (-3 ... 3) does
344                                          * the ISO year begin on?
345                                          */
346                                         bot = ((yday + 11 - wday) %
347                                             DAYSPERWEEK) - 3;
348                                         /*
349                                          * What yday does the NEXT
350                                          * ISO year begin on?
351                                          */
352                                         top = bot - (len % DAYSPERWEEK);
353                                         if (top < -3)
354                                                 top += DAYSPERWEEK;
355                                         top += len;
356                                         if (yday >= top) {
357                                                 ++base;
358                                                 w = 1;
359                                                 break;
360                                         }
361                                         if (yday >= bot) {
362                                                 w = 1 + ((yday - bot) /
363                                                     DAYSPERWEEK);
364                                                 break;
365                                         }
366                                         --base;
367                                         yday += isleap_sum(year, base) ?
368                                             DAYSPERLYEAR : DAYSPERNYEAR;
369                                 }
370 #ifdef XPG4_1994_04_09
371                                 if ((w == 52 && t->tm_mon == TM_JANUARY) ||
372                                     (w == 1 && t->tm_mon == TM_DECEMBER))
373                                         w = 53;
374 #endif /* defined XPG4_1994_04_09 */
375                                 if (*format == 'V')
376                                         pt = _conv(w,
```

```
377                                     PADDING(PAD_FMT_WEEKOFYEAR),
378                                     pt, ptlim);
379                             else if (*format == 'g') {
380                                     pt = _yconv(year, base, 0, 1,
381                                         pt, ptlim);
382                             } else
383                                     pt = _yconv(year, base, 1, 1,
384                                         pt, ptlim);
385                     }
386                             continue;
387                     case 'v':
388                             /*
389                              * From Arnold Robbins' strftime version 3.0:
390                              * "date as dd-bbb-YYYY"
391                              * (ado, 1993-05-24)
392                              */
393                             pt = _fmt("%e-%b-%Y", t, pt, ptlim,
394                                 warnp, loc);
333                             pt = _fmt("%e-%b-%Y", t, pt, ptlim);
395                             continue;
396                     case 'W':
397                             pt = _conv((t->tm_yday + DAYSPERWEEK -
398                                 (t->tm_wday ?
399                                 (t->tm_wday - 1) :
400                                 (DAYSPERWEEK - 1))) / DAYSPERWEEK,
401                                 PADDING(PAD_FMT_WEEKOFYEAR),
402                                 pt, ptlim);
403                             continue;
404                     case 'w':
405                             pt = _conv(t->tm_wday, "%d", pt, ptlim);
406                             continue;
407                     case 'X':
408                             pt = _fmt(tptr->X_fmt, t, pt, ptlim,
409                                 warnp, loc);
347                             pt = _fmt(tptr->X_fmt, t, pt, ptlim);
410                             continue;
411                     case 'x':
412                     {
413                             int     warn2 = IN_SOME;

415                             pt = _fmt(tptr->x_fmt, t, pt, ptlim,
416                                 &warn2, loc);
417                     }
350                             pt = _fmt(tptr->x_fmt, t, pt, ptlim);
418                             continue;
419                     case 'y':
420                             pt = _yconv(t->tm_year, TM_YEAR_BASE, 0, 1,
421                                 pt, ptlim);
422                             continue;
423                     case 'Y':
424                             pt = _yconv(t->tm_year, TM_YEAR_BASE, 1, 1,
425                                 pt, ptlim);
426                             continue;
427                     case 'Z':
428                             if (t->tm_isdst >= 0)
429                                     pt = _add(tzname[t->tm_isdst != 0],
430                                         pt, ptlim);
431                             /*
432                              * C99 says that %Z must be replaced by the
433                              * empty string if the time zone is not
434                              * determinable.
435                              */
436                             continue;
437                     case 'z':
438                             {
439                             int             diff;
```



```
440                             char const *    sign;

442                             if (t->tm_isdst < 0)
443                                     continue;
444                             /*
445                              * C99 says that the UTC offset must
446                              * be computed by looking only at
447                              * tm_isdst. This requirement is
448                              * incorrect, since it means the code
449                              * must rely on magic (in this case
450                              * altzone and timezone), and the
451                              * magic might not have the correct
452                              * offset. Doing things correctly is
453                              * tricky and requires disobeying C99;
454                              * see GNU C strftime for details.
455                              * For now, punt and conform to the
456                              * standard, even though it's incorrect.
457                              *
458                              * C99 says that %z must be replaced by the
459                              * empty string if the time zone is not
460                              * determinable, so output nothing if the
461                              * appropriate variables are not available.
462                              */
463                             if (t->tm_isdst == 0)
464                                     diff = -timezone;
465                             else
466                                     diff = -altzone;
467                             if (diff < 0) {
468                                     sign = "-";
469                                     diff = -diff;
470                             } else
471                                     sign = "+";
472                             pt = _add(sign, pt, ptlim);
473                             diff /= SECSPERMIN;
474                             diff = (diff / MINSPERHOUR) * 100 +
475                                 (diff % MINSPERHOUR);
476                             pt = _conv(diff, PADDING(PAD_FMT_YEAR),
477                                 pt, ptlim);
478                             }
479                             continue;
480                     case '+':
481                             pt = _fmt(tptr->date_fmt, t, pt, ptlim,
482                                 warnp, loc);
414                             pt = _fmt(tptr->date_fmt, t, pt, ptlim);
483                             continue;
484                     case '-':
485                             if (PadIndex != PAD_DEFAULT)
486                                     break;
487                             PadIndex = PAD_LESS;
488                             goto label;
489                     case '_':
490                             if (PadIndex != PAD_DEFAULT)
491                                     break;
492                             PadIndex = PAD_SPACE;
493                             goto label;
494                     case '0':
495                             if (PadIndex != PAD_DEFAULT)
496                                     break;
497                             PadIndex = PAD_ZERO;
498                             goto label;
499                     case '%':
500                     /*
501                      * X311J/88-090 (4.12.3.5): if conversion char is
502                      * undefined, behavior is undefined. Print out the
503                      * character itself as printf(3) also does.
504                      */
```

```
 505                         default:
 506                                 break;
 507                         }
 508                 }
 509                 if (pt == ptlim)
 510                         break;
 511                 *pt++ = *format;
 512         }
 513         return (pt);
 514 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   11856 Thu Jul 11 12:26:14 2013
new/usr/src/lib/libc/port/locale/strptime.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2011, Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1994 Powerdog Industries.  All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 * under sponsorship from the FreeBSD Foundation.
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer
 *    in the documentation and/or other materials provided with the
 *    distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY POWERDOG INDUSTRIES ``AS IS'' AND ANY
 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE POWERDOG INDUSTRIES 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 USE OF THIS SOFTWARE,
 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * The views and conclusions contained in the software and documentation
 * are those of the authors and should not be interpreted as representing
 * official policies, either expressed or implied, of Powerdog Industries.
 */

#include "lint.h"
#include <time.h>
#include <ctype.h>
#include <errno.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <alloca.h>
#include "timelocal.h"

#define asizeof(a)      (sizeof (a) / sizeof ((a)[0]))

#define F_GMT           (1 << 0)
#define F_ZERO          (1 << 1)
#define F_RECURSE       (1 << 2)

static char *
__strptime(const char *buf, const char *fmt, struct tm *tm, int *flagsp,
    int *GMTp, locale_t locale)
__strptime(const char *buf, const char *fmt, struct tm *tm, int *flagsp)
{
        char    c;
```



```
        const char *ptr;
        int     i, len, recurse = 0;
        int Ealternative, Oalternative;
        struct lc_time_T *tptr = __get_current_time_locale(locale);
        struct lc_time_T *tptr = __get_current_time_locale();

        if (*flagsp & F_RECURSE)
                recurse = 1;
        *flagsp |= F_RECURSE;

        if (*flagsp & F_ZERO)
                (void) memset(tm, 0, sizeof (*tm));
        *flagsp &= ~F_ZERO;

        ptr = fmt;
        while (*ptr != 0) {
                if (*buf == 0)
                        break;

                c = *ptr++;

                if (c != '%') {
                        if (isspace(c))
                                while (isspace(*buf))
                                        buf++;
                        else if (c != *buf++)
                                return (NULL);
                        continue;
                }

                Ealternative = 0;
                Oalternative = 0;
label:
                c = *ptr++;
                switch (c) {
                case 0:
                case '%':
                        if (*buf++ != '%')
                                return (NULL);
                        break;

                case '+':
                        buf = __strptime(buf, tptr->date_fmt, tm, flagsp,
                            GMTp, locale);
                        buf = __strptime(buf, tptr->date_fmt, tm, flagsp);
                        if (buf == NULL)
                                return (NULL);
                        break;

                case 'C':
                        if (!isdigit(*buf))
                                return (NULL);

                        /* XXX This will break for 3-digit centuries. */
                        len = 2;
                        for (i = 0; len && isdigit(*buf); buf++) {
                                i *= 10;
                                i += *buf - '0';
                                len--;
                        }
                        if (i < 19)
                                return (NULL);

                        tm->tm_year = i * 100 - 1900;
                        break;
```

```
125                 case 'c':
126                         buf = __strptime(buf, tptr->c_fmt, tm, flagsp,
127                             GMTp, locale);
 72                         buf = __strptime(buf, tptr->c_fmt, tm, flagsp);
128                         if (buf == NULL)
129                                 return (NULL);
130                         break;

132                 case 'D':
133                         buf = __strptime(buf, "%m/%d/%y", tm, flagsp,
134                             GMTp, locale);
 78                         buf = __strptime(buf, "%m/%d/%y", tm, flagsp);
135                         if (buf == NULL)
136                                 return (NULL);
137                         break;

139                 case 'E':
140                         if (Ealternative || Oalternative)
141                                 break;
142                         Ealternative++;
143                         goto label;

145                 case 'O':
146                         if (Ealternative || Oalternative)
147                                 break;
148                         Oalternative++;
149                         goto label;

151                 case 'F':
152                         buf = __strptime(buf, "%Y-%m-%d", tm, flagsp,
153                             GMTp, locale);
 96                         buf = __strptime(buf, "%Y-%m-%d", tm, flagsp);
154                         if (buf == NULL)
155                                 return (NULL);
156                         break;

158                 case 'R':
159                         buf = __strptime(buf, "%H:%M", tm, flagsp,
160                             GMTp, locale);
102                         buf = __strptime(buf, "%H:%M", tm, flagsp);
161                         if (buf == NULL)
162                                 return (NULL);
163                         break;

165                 case 'r':
166                         buf = __strptime(buf, tptr->ampm_fmt, tm, flagsp,
167                             GMTp, locale);
108                         buf = __strptime(buf, tptr->ampm_fmt, tm, flagsp);
168                         if (buf == NULL)
169                                 return (NULL);
170                         break;

172                 case 'T':
173                         buf = __strptime(buf, "%H:%M:%S", tm, flagsp,
174                             GMTp, locale);
114                         buf = __strptime(buf, "%H:%M:%S", tm, flagsp);
175                         if (buf == NULL)
176                                 return (NULL);
177                         break;

179                 case 'X':
180                         buf = __strptime(buf, tptr->X_fmt, tm, flagsp,
181                             GMTp, locale);
120                         buf = __strptime(buf, tptr->X_fmt, tm, flagsp);
182                         if (buf == NULL)
183                                 return (NULL);
```



```
184                         break;

186                 case 'x':
187                         buf = __strptime(buf, tptr->x_fmt, tm, flagsp,
188                             GMTp, locale);
126                         buf = __strptime(buf, tptr->x_fmt, tm, flagsp);
189                         if (buf == NULL)
190                                 return (NULL);
191                         break;

193                 case 'j':
194                         if (!isdigit(*buf))
195                                 return (NULL);

197                         len = 3;
198                         for (i = 0; len && isdigit(*buf); buf++) {
199                                 i *= 10;
200                                 i += *buf - '0';
201                                 len--;
202                         }
203                         if (i < 1 || i > 366)
204                                 return (NULL);

206                         tm->tm_yday = i - 1;
207                         break;

209                 case 'M':
210                 case 'S':
211                         if (*buf == 0 || isspace(*buf))
212                                 break;

214                         if (!isdigit(*buf))
215                                 return (NULL);

217                         len = 2;
218                         for (i = 0; len && isdigit(*buf); buf++) {
219                                 i *= 10;
220                                 i += *buf - '0';
221                                 len--;
222                         }

224                         if (c == 'M') {
225                                 if (i > 59)
226                                         return (NULL);
227                                 tm->tm_min = i;
228                         } else {
229                                 if (i > 60)
230                                         return (NULL);
231                                 tm->tm_sec = i;
232                         }

234                         if (isspace(*buf))
235                                 while (*ptr != 0 && !isspace(*ptr))
236                                         ptr++;
237                         break;

239                 case 'H':
240                 case 'I':
241                 case 'k':
242                 case 'l':
243                         /*
244                          * Of these, %l is the only specifier explicitly
245                          * documented as not being zero-padded.  However,
246                          * there is no harm in allowing zero-padding.
247                          *
248                          * XXX The %l specifier may gobble one too many
```

```
249                          * digits if used incorrectly.
250                          */
251                         if (!isdigit(*buf))
252                                 return (NULL);
253
254                         len = 2;
255                         for (i = 0; len && isdigit(*buf); buf++) {
256                                 i *= 10;
257                                 i += *buf - '0';
258                                 len--;
259                         }
260                         if (c == 'H' || c == 'k') {
261                                 if (i > 23)
262                                         return (NULL);
263                         } else if (i > 12)
264                                 return (NULL);
265
266                         tm->tm_hour = i;
267
268                         if (isspace(*buf))
269                                 while (*ptr != 0 && !isspace(*ptr))
270                                         ptr++;
271                         break;
272
273                 case 'p':
274                         /*
275                          * XXX This is bogus if parsed before hour-related
276                          * specifiers.
277                          */
278                         len = strlen(tptr->am);
279                         if (strncasecmp(buf, tptr->am, len) == 0) {
280                                 if (tm->tm_hour > 12)
281                                         return (NULL);
282                                 if (tm->tm_hour == 12)
283                                         tm->tm_hour = 0;
284                                 buf += len;
285                                 break;
286                         }
287
288                         len = strlen(tptr->pm);
289                         if (strncasecmp(buf, tptr->pm, len) == 0) {
290                                 if (tm->tm_hour > 12)
291                                         return (NULL);
292                                 if (tm->tm_hour != 12)
293                                         tm->tm_hour += 12;
294                                 buf += len;
295                                 break;
296                         }
297
298                         return (NULL);
299
300                 case 'A':
301                 case 'a':
302                         for (i = 0; i < asizeof(tptr->weekday); i++) {
303                                 len = strlen(tptr->weekday[i]);
304                                 if (strncasecmp(buf, tptr->weekday[i], len) ==
305                                     0)
306                                         break;
307                                 len = strlen(tptr->wday[i]);
308                                 if (strncasecmp(buf, tptr->wday[i], len) == 0)
309                                         break;
310                         }
311                         if (i == asizeof(tptr->weekday))
312                                 return (NULL);
313
314                         tm->tm_wday = i;
```



```
315                         buf += len;
316                         break;
317
318                 case 'U':
319                 case 'W':
320                         /*
321                          * XXX This is bogus, as we can not assume any valid
322                          * information present in the tm structure at this
323                          * point to calculate a real value, so just check the
324                          * range for now.
325                          */
326                         if (!isdigit(*buf))
327                                 return (NULL);
328
329                         len = 2;
330                         for (i = 0; len && isdigit(*buf); buf++) {
331                                 i *= 10;
332                                 i += *buf - '0';
333                                 len--;
334                         }
335                         if (i > 53)
336                                 return (NULL);
337
338                         if (isspace(*buf))
339                                 while (*ptr != 0 && !isspace(*ptr))
340                                         ptr++;
341                         break;
342
343                 case 'w':
344                         if (!isdigit(*buf))
345                                 return (NULL);
346
347                         i = *buf - '0';
348                         if (i > 6)
349                                 return (NULL);
350
351                         tm->tm_wday = i;
352
353                         if (isspace(*buf))
354                                 while (*ptr != 0 && !isspace(*ptr))
355                                         ptr++;
356                         break;
357
358                 case 'e':
359                         /*
360                          * The %e format has a space before single digits
361                          * which we need to skip.
362                          */
363                         if (isspace(*buf))
364                                 buf++;
365                         /* FALLTHROUGH */
366                 case 'd':
367                         /*
368                          * The %e specifier is explicitly documented as not
369                          * being zero-padded but there is no harm in allowing
370                          * such padding.
371                          *
372                          * XXX The %e specifier may gobble one too many
373                          * digits if used incorrectly.
374                          */
375                         if (!isdigit(*buf))
376                                 return (NULL);
377
378                         len = 2;
379                         for (i = 0; len && isdigit(*buf); buf++) {
380                                 i *= 10;
```

```
381                             i += *buf - '0';
382                             len--;
383                     }
384                     if (i > 31)
385                             return (NULL);

387                     tm->tm_mday = i;

389                     if (isspace(*buf))
390                             while (*ptr != 0 && !isspace(*ptr))
391                                     ptr++;
392                     break;

394             case 'B':
395             case 'b':
396             case 'h':
397                     for (i = 0; i < asizeof(tptr->month); i++) {
398                             len = strlen(tptr->month[i]);
399                             if (strncasecmp(buf, tptr->month[i], len) == 0)
400                                     break;
401                     }
402                     /*
403                      * Try the abbreviated month name if the full name
404                      * wasn't found.
405                      */
406                     if (i == asizeof(tptr->month)) {
407                             for (i = 0; i < asizeof(tptr->month); i++) {
408                                     len = strlen(tptr->mon[i]);
409                                     if (strncasecmp(buf, tptr->mon[i],
410                                         len) == 0)
411                                             break;
412                             }
413                     }
414                     if (i == asizeof(tptr->month))
415                             return (NULL);

417                     tm->tm_mon = i;
418                     buf += len;
419                     break;

421             case 'm':
422                     if (!isdigit(*buf))
423                             return (NULL);

425                     len = 2;
426                     for (i = 0; len && isdigit(*buf); buf++) {
427                             i *= 10;
428                             i += *buf - '0';
429                             len--;
430                     }
431                     if (i < 1 || i > 12)
432                             return (NULL);

434                     tm->tm_mon = i - 1;

436                     if (isspace(*buf))
437                             while (*ptr != NULL && !isspace(*ptr))
438                                     ptr++;
439                     break;

441             case 's':
442                     {
443                     char *cp;
444                     int sverrno;
445                     time_t t;
```



```
447                             sverrno = errno;
448                             errno = 0;
449                             t = strtol(buf, &cp, 10);
450                             if (errno == ERANGE) {
451                                     errno = sverrno;
452                                     return (NULL);
453                             }
454                             errno = sverrno;
455                             buf = cp;
456                             (void) gmtime_r(&t, tm);
457                             *flagsp |= F_GMT;
458                             }
459                             break;

461             case 'Y':
462             case 'y':
463                             if (*buf == NULL || isspace(*buf))
464                                     break;

466                             if (!isdigit(*buf))
467                                     return (NULL);

469                             len = (c == 'Y') ? 4 : 2;
470                             for (i = 0; len && isdigit(*buf); buf++) {
471                                     i *= 10;
472                                     i += *buf - '0';
473                                     len--;
474                             }
475                             if (c == 'Y')
476                                     i -= 1900;
477                             if (c == 'y' && i < 69)
478                                     i += 100;
479                             if (i < 0)
480                                     return (NULL);

482                             tm->tm_year = i;

484                             if (isspace(*buf))
485                                     while (*ptr != 0 && !isspace(*ptr))
486                                             ptr++;
487                             break;

489             case 'Z':
490                             {
491                             const char *cp = buf;
492                             char *zonestr;

494                             while (isupper(*cp))
495                                     ++cp;
496                             if (cp - buf) {
497                                     zonestr = alloca(cp - buf + 1);
498                                     (void) strncpy(zonestr, buf, cp - buf);
499                                     zonestr[cp - buf] = '\0';
500                                     tzset();
501                                     if (strcmp(zonestr, "GMT") == 0) {
502                                             *flagsp |= F_GMT;
503                                     } else if (0 == strcmp(zonestr, tzname[0])) {
504                                             tm->tm_isdst = 0;
505                                     } else if (0 == strcmp(zonestr, tzname[1])) {
506                                             tm->tm_isdst = 1;
507                                     } else {
508                                             return (NULL);
509                                     }
510                                     buf += cp - buf;
511                             }
512                             }
```

```
513                         break;

515                 case 'z':
516                         {
517                         int sign = 1;

519                         if (*buf != '+') {
520                                 if (*buf == '-')
521                                         sign = -1;
522                                 else
523                                         return (NULL);
524                         }
525                         buf++;
526                         i = 0;
527                         for (len = 4; len > 0; len--) {
528                                 if (!isdigit(*buf))
529                                         return (NULL);
530                                 i *= 10;
531                                 i += *buf - '0';
532                                 buf++;
533                         }

535                         tm->tm_hour -= sign * (i / 100);
536                         tm->tm_min -= sign * (i % 100);
537                         *flagsp |= F_GMT;
538                         }
539                         break;
540                 }
541         }

543         if (!recurse) {
544                 if (buf && (*flagsp & F_GMT)) {
545                         time_t t = timegm(tm);
546                         (void) localtime_r(&t, tm);
547                 }
548         }

550         return ((char *)buf);
551 }

553 char *
554 strptime_l(const char * __restrict buf, const char * __restrict fmt,
555     struct tm * __restrict tm, locale_t loc)
556 {
557         char *ret;
558         int gmt;
559         int flags = F_ZERO;
560         FIX_LOCALE(loc);

562         gmt = 0;
563         ret = __strptime(buf, fmt, tm, &flags, &gmt, loc);
564         if (ret && gmt) {
565                 time_t t = timegm(tm);
566                 localtime_r(&t, tm);
567         }

569         return (ret);
570 }

572 char *
573 #endif /* ! codereview */
574 strptime(const char *buf, const char *fmt, struct tm *tm)
575 {
576         int     flags = F_ZERO;

578         return (strptime_l(buf, fmt, tm, __get_locale()));
```



```
579         // XXX return (__strptime(buf, fmt, tm, &flags));
492         return (__strptime(buf, fmt, tm, &flags));
580 }

582 /*
583  * This is used by Solaris, and is a variant that does not clear the
584  * incoming tm.  It is triggered by -D_STRPTIME_DONTZERO.
585  */
586 char *
587 __strptime_dontzero(const char *buf, const char *fmt, struct tm *tm)
588 {
589         int     flags = 0;
590         int     gmt = 0;
591 #endif /* ! codereview */

593         return (__strptime(buf, fmt, tm, &flags, &gmt, __get_locale()));
594         /* XXX */
503         return (__strptime(buf, fmt, tm, &flags));
595 }
______unchanged_portion_omitted_
```

```
**********************************************************
    3783 Thu Jul 11 12:26:14 2013
new/usr/src/lib/libc/port/locale/timelocal.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 2001 Alexey Zelkin <phantom@FreeBSD.org>
 * Copyright (c) 1997 FreeBSD Inc.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "lint.h"
#include <stddef.h>
#include "ldpart.h"
#include "timelocal.h"

static struct lc_time_T _time_locale;
static int _time_using_locale;
static char *time_locale_buf;

struct xlocale_time __xlocale_global_time;

#endif /* ! codereview */
#define LCTIME_SIZE (sizeof (struct lc_time_T) / sizeof (char *))

static const struct lc_time_T   _C_time_locale = {
        {
                "Jan", "Feb", "Mar", "Apr", "May", "Jun",
                "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
        }, {
                "January", "February", "March", "April", "May", "June",
                "July", "August", "September", "October", "November", "December"
        }, {
                "Sun", "Mon", "Tue", "Wed",
                "Thu", "Fri", "Sat"
        }, {
                "Sunday", "Monday", "Tuesday", "Wednesday",
                "Thursday", "Friday", "Saturday"
        },

        /* X_fmt */
        "%H:%M:%S",

        /*
```



```
         * x_fmt
         * Since the C language standard calls for
         * "date, using locale's date format," anything goes.
         * Using just numbers (as here) makes Quakers happier;
         * it's also compatible with SVR4.
         */
        "%m/%d/%y",

        /*
         * c_fmt
         */
        "%a %b %e %H:%M:%S %Y",

        /* am */
        "AM",

        /* pm */
        "PM",

        /* date_fmt */
        "%a %b %e %H:%M:%S %Z %Y",

        /*
         * ampm_fmt - To determine 12-hour clock format time (empty, if N/A)
         */
        "%I:%M:%S %p"
};

static void
destruct_time(void *v)
{
        struct xlocale_time *l = v;

        if (l->buffer != NULL)
                free(l->buffer);

        free(l);
}

#endif /* ! codereview */
struct lc_time_T *
__get_current_time_locale(locale_t loc)
__get_current_time_locale(void)
{
        return (loc->using_time_locale
            ? &((struct xlocale_time *)loc->components[XLC_TIME])->locale
            : (struct lc_time_T *)&_C_time_locale);
}

static int
time_load_locale(struct xlocale_time *l, int *using_locale, const char *name)
{
        struct lc_time_T *time_locale = &l->locale;

        return (__part_load_locale(name, using_locale,
            &l->buffer, "LC_TIME",
            LCTIME_SIZE, LCTIME_SIZE,
            (const char **)time_locale));
        return (_time_using_locale ? &_time_locale :
            (struct lc_time_T *)&_C_time_locale);
}

int
__time_load_locale(const char *name)
{
        return (time_load_locale(&__xlocale_global_time,
```

```
125             &__xlocale_global_locale.using_time_locale, name));
126 }

128 void *
129 __time_load(const char* name, locale_t loc)
130 {
131         struct xlocale_time *new;

133         new = calloc(sizeof(struct xlocale_time), 1);
134         if (new == NULL)
135                 return (NULL);

137         new->header.header.destructor = destruct_time;
138         if (time_load_locale(new, &loc->using_time_locale, name) ==
139             _LDP_ERROR) {
140                 xlocale_release(new);
141                 return (NULL);
142         }

144         return (new);
 47         return (__part_load_locale(name, &_time_using_locale,
 48             &time_locale_buf, "LC_TIME", LCTIME_SIZE, LCTIME_SIZE,
 49             (const char **)&_time_locale));
145 }
```

_____*unchanged_portion_omitted_*

```
**********************************************************
    2196 Thu Jul 11 12:26:15 2013
new/usr/src/lib/libc/port/locale/timelocal.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1997-2002 FreeBSD Project.
 * All rights reserved.
 *
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 * Portions of this software were developed by David Chisnall
 *
#endif /* ! codereview */
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _TIMELOCAL_H_
#define _TIMELOCAL_H_

#include "xlocale_private.h"

#endif /* ! codereview */
/*
 * Private header file for the strftime and strptime localization
 * stuff.
 */
struct lc_time_T {
        const char      *mon[12];
        const char      *month[12];
        const char      *wday[7];
        const char      *weekday[7];
        const char      *X_fmt;
        const char      *x_fmt;
        const char      *c_fmt;
        const char      *am;
        const char      *pm;
        const char      *date_fmt;
        const char      *ampm_fmt;
};

struct xlocale_time {
        struct xlocale_component header;
        char *buffer;
        struct lc_time_T locale;
};
```



```
struct lc_time_T *__get_current_time_locale(locale_t);
struct lc_time_T *__get_current_time_locale(void);
int     __time_load_locale(const char *);

#endif /* !_TIMELOCAL_H_ */
```

```
**********************************************************
    2880 Thu Jul 11 12:26:15 2013
new/usr/src/lib/libc/port/locale/wcwidth.c
2964 need POSIX 2008 locale object support
**********************************************************
______unchanged_portion_omitted_

  66 #pragma weak _scrwidth = scrwidth

  68 int
  69 wcwidth(wchar_t wc, locale_t locale)
  70 {
  71         /* XXX */
  72         return (0);
  73 }

  75 #endif /* ! codereview */
  76 /*
  77  * This is a Solaris extension.  It never returns a negative width, even for
  78  * non-printable characters.  It is used internally by the printf
  79  * implementation for %ws.
  80  */
  81 int
  82 scrwidth(wchar_t wc)
  83 {
  84         int     v = wcwidth(wc);
  85         return (v > 0 ? v : 0);
  86 }
```

```
**********************************************************
    8573 Thu Jul 11 12:26:15 2013
new/usr/src/lib/libc/port/locale/xlocale.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 *
 * This software was developed by David Chisnall under sponsorship from
 * the FreeBSD Foundation.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include <pthread.h>
#include <stdio.h>
#include <string.h>

#include "runetype.h"
#include "xlocale_private.h"

/*
 * Each locale loader declares a global component.  This is used by setlocale()
 * and also by xlocale with LC_GLOBAL_LOCALE..
 */
extern struct xlocale_component __xlocale_global_collate;
extern struct xlocale_component __xlocale_global_ctype;
extern struct xlocale_component __xlocale_global_monetary;
extern struct xlocale_component __xlocale_global_numeric;
extern struct xlocale_component __xlocale_global_time;
extern struct xlocale_component __xlocale_global_messages;

/*
 * And another version for the statically-allocated C locale.  We only have
 * components for the parts that are expected to be sensible.
 */
extern struct xlocale_component __xlocale_C_collate;
extern struct xlocale_component __xlocale_C_ctype;

/*
 * The locale for this thread.
 */
__thread locale_t __thread_locale;

/*
 * Flag indicating that one or more per-thread locales exist.
```



```
 */
int __has_thread_locale;

struct _xlocale __xlocale_global_locale = {
        {0},
        {
                &__xlocale_global_collate,
                &__xlocale_global_ctype,
                &__xlocale_global_monetary,
                &__xlocale_global_numeric,
                &__xlocale_global_time,
                &__xlocale_global_messages
        },
        1,
        0,
        1,
        0
};

struct _xlocale __xlocale_C_locale = {
        {0},
        {
                &__xlocale_C_collate,
                &__xlocale_C_ctype,
                0, 0, 0, 0
        },
        1,
        0,
        1,
        0
};

static void*(*constructors[])(const char*, locale_t) =
{
        __collate_load,
        __ctype_load,
        __monetary_load,
        __numeric_load,
        __time_load,
        __messages_load
};

static pthread_key_t locale_info_key;
static int fake_tls;
static locale_t thread_local_locale;

static void
init_key(void)
{
        pthread_key_create(&locale_info_key, xlocale_release);
        pthread_setspecific(locale_info_key, (void*)42);
        if (pthread_getspecific(locale_info_key) == (void*)42) {
                pthread_setspecific(locale_info_key, 0);
        } else {
                fake_tls = 1;
        }
        /* At least one per-thread locale has now been set. */
        __has_thread_locale = 1;
        __detect_path_locale();
}

static pthread_once_t once_control = PTHREAD_ONCE_INIT;

static locale_t
get_thread_locale(void)
{
```

```
129         // XXX  _once(&once_control, init_key);
130         (void) pthread_once(&once_control, init_key);
131 
132         return (fake_tls ? thread_local_locale :
133             pthread_getspecific(locale_info_key));
134 }
135 
136 static void
137 set_thread_locale(locale_t loc)
138 {
139 
140         // XXX _once(&once_control, init_key);
141         (void) pthread_once(&once_control, init_key);
142 
143         if (NULL != loc) {
144                 xlocale_retain((struct xlocale_refcounted*)loc);
145         }
146         locale_t old = pthread_getspecific(locale_info_key);
147         if ((NULL != old) && (loc != old)) {
148                 xlocale_release((struct xlocale_refcounted*)old);
149         }
150         if (fake_tls) {
151                 thread_local_locale = loc;
152         } else {
153                 pthread_setspecific(locale_info_key, loc);
154         }
155         __thread_locale = loc;
156         __set_thread_rune_locale(loc);
157 }
158 
159 /*
160  * Clean up a locale, once its reference count reaches zero.  This function is
161  * called by xlocale_release(), it should not be called directly.
162  */
163 static void
164 destruct_locale(void *l)
165 {
166         locale_t loc = l;
167 
168         for (int type = 0; type < XLC_LAST; type++) {
169                 if (loc->components[type]) {
170                         xlocale_release(loc->components[type]);
171                 }
172         }
173         if (loc->csym) {
174                 free(loc->csym);
175         }
176         free(l);
177 }
178 
179 /*
180  * Allocates a new, uninitialised, locale.
181  */
182 static locale_t
183 alloc_locale(void)
184 {
185         locale_t new = calloc(sizeof(struct _xlocale), 1);
186 
187         new->header.destructor = destruct_locale;
188         new->monetary_locale_changed = 1;
189         new->numeric_locale_changed = 1;
190         return (new);
191 }
192 
193 static void
```



```
194 copyflags(locale_t new, locale_t old)
195 {
196         new->using_monetary_locale = old->using_monetary_locale;
197         new->using_numeric_locale = old->using_numeric_locale;
198         new->using_time_locale = old->using_time_locale;
199         new->using_messages_locale = old->using_messages_locale;
200 }
201 
202 static int
203 dupcomponent(int type, locale_t base, locale_t new)
204 {
205         /*
206          * Always copy from the global locale, since it has mutable components.
207          */
208         struct xlocale_component *src = base->components[type];
209 
210         if (&__xlocale_global_locale == base) {
211                 new->components[type] = constructors[type](src->locale, new);
212                 if (new->components[type]) {
213                         strncpy(new->components[type]->locale, src->locale,
214                             ENCODING_LEN);
215                 }
216         } else if (base->components[type]) {
217                 new->components[type] = xlocale_retain(base->components[type]);
218         } else {
219                 /*
220                  * If the component was NULL, return success - if base is a
221                  * valid locale then the flag indicating that this isn't
222                  * present should be set.  If it isn't a valid locale, then
223                  * we're stuck anyway.
224                  */
225                 return (1);
226         }
227         return (0 != new->components[type]);
228 }
229 
230 /*
231  * Public interfaces.  These are the five public functions described by the
232  * xlocale interface.
233  */
234 locale_t
235 newlocale(int mask, const char *locale, locale_t base)
236 {
237         int type;
238         const char *realLocale = locale;
239         int useenv = 0;
240         int success = 1;
241 
242         // XXX _once(&once_control, init_key);
243         (void) pthread_once(&once_control, init_key);
244 
245         locale_t new = alloc_locale();
246         if (NULL == new) {
247                 return (NULL);
248         }
249 
250         FIX_LOCALE(base);
251         copyflags(new, base);
252 
253         if (NULL == locale) {
254                 realLocale = "C";
255         } else if ('\0' == locale[0]) {
256                 useenv = 1;
257         }
258 
259         for (type = 0; type < XLC_LAST; type++) {
```

```
260                 if (mask & 1) {
261                         if (useenv) {
262                                 realLocale = __get_locale_env(type);
263                         }
264                         new->components[type] =
265                              constructors[type](realLocale, new);
266                         if (new->components[type]) {
267                                 strncpy(new->components[type]->locale,
268                                      realLocale, ENCODING_LEN);
269                         } else {
270                                 success = 0;
271                                 break;
272                         }
273                 } else {
274                         if (!dupcomponent(type, base, new)) {
275                                 success = 0;
276                                 break;
277                         }
278                 }
279                 mask >>= 1;
280         }
281         if (0 == success) {
282                 xlocale_release(new);
283                 new = NULL;
284         }

286         return (new);
287 }

289 locale_t
290 duplocale(locale_t base)
291 {
292         locale_t new = alloc_locale();
293         int type;

295         // XXX _once(&once_control, init_key);
296         (void) pthread_once(&once_control, init_key);

298         if (NULL == new) {
299                 return (NULL);
300         }
301
302         FIX_LOCALE(base);
303         copyflags(new, base);

305         for (type=0 ; type<XLC_LAST ; type++) {
306                 dupcomponent(type, base, new);
307         }

309         return (new);
310 }

312 /*
313  * Free a locale_t.  This is quite a poorly named function.  It actually
314  * disclaims a reference to a locale_t, rather than freeing it.
315  */
316 int
317 freelocale(locale_t loc)
318 {
319         /* Fail if we're passed something that isn't a locale. */
320         if ((NULL == loc) || (LC_GLOBAL_LOCALE == loc)) {
321                 return (-1);
322         }
323         /*
324          * If we're passed the global locale, pretend that we freed it but don't
325          * actually do anything.
```



```
326          */
327         if (&__xlocale_global_locale == loc) {
328                 return (0);
329         }
330         xlocale_release(loc);
331         return (0);
332 }

334 /*
335  * Returns the name of the locale for a particular component of a locale_t.
336  */
337 const char *
338 querylocale(int mask, locale_t loc)
339 {
340         int type = ffs(mask) - 1;
341         FIX_LOCALE(loc);
342         if (type >= XLC_LAST)
343                 return (NULL);
344         if (loc->components[type])
345                 return (loc->components[type]->locale);
346         return ("C");
347 }

349 /*
350  * Installs the specified locale_t as this thread's locale.
351  */
352 locale_t
353 uselocale(locale_t loc)
354 {
355         locale_t old = get_thread_locale();
356         if (NULL != loc) {
357                 if (LC_GLOBAL_LOCALE == loc) {
358                         loc = NULL;
359                 }
360                 set_thread_locale(loc);
361         }
362         return (old ? old : LC_GLOBAL_LOCALE);
363 }
364 #endif /* ! codereview */
```

```
**********************************************************
    8038 Thu Jul 11 12:26:15 2013
new/usr/src/lib/libc/port/locale/xlocale_private.h
2964 need POSIX 2008 locale object support
**********************************************************
```

```
/*
 * Copyright (c) 2011 The FreeBSD Foundation
 * All rights reserved.
 *
 * This software was developed by David Chisnall under sponsorship from
 * the FreeBSD Foundation.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS 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 USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifndef _XLOCALE_PRIVATE__H_
#define _XLOCALE_PRIVATE__H_

//#include <xlocale.h> /* XXX */
#include <locale.h>
#include <stdlib.h>
#include <stdint.h>
#include <sys/types.h>
#include "setlocale.h"
#include <wchar_impl.h> /* XXX */

enum {
        XLC_COLLATE = 0,
        XLC_CTYPE,
        XLC_MONETARY,
        XLC_NUMERIC,
        XLC_TIME,
        XLC_MESSAGES,
        XLC_LAST
};


/*
 * Header used for objects that are reference counted.  Objects may optionally
 * have a destructor associated, which is responsible for destroying the
 * structure.  Global / static versions of the structure should have no
 * destructor set - they can then have their reference counts manipulated as
 * normal, but will not do anything with them.
 *
 * The header stores a retain count - objects are assumed to have a reference
 * count of 1 when they are created, but the retain count is 0.  When the
 * retain count is less than 0, they are freed.
```



```
 */
struct xlocale_refcounted {
        /* Number of references to this component. */
        long retain_count;
        /* Function used to destroy this component, if one is required */
        void(*destructor)(void*);
};

/*
 * Header for a locale component.  All locale components must begin with this
 * header.
 */
struct xlocale_component {
        struct xlocale_refcounted header;
        /* Name of the locale used for this component. */
        char locale[ENCODING_LEN+1];
};

/*
 * xlocale structure, stores per-thread locale information.
 */
struct _xlocale {
        struct xlocale_refcounted header;
        /* Components for the locale. */
        struct xlocale_component *components[XLC_LAST];
        /*
         * Flag indicating if components[XLC_MONETARY] has changed since the
         * last call to localeconv_l() with this locale.
         */
        int monetary_locale_changed;
        /** Flag indicating whether this locale is actually using a locale for
         * LC_MONETARY (1), or if it should use the C default instead (0). */
        int using_monetary_locale;
        /** Flag indicating if components[XLC_NUMERIC] has changed since the
         * last call to localeconv_l() with this locale. */
        int numeric_locale_changed;
        /** Flag indicating whether this locale is actually using a locale for
         * LC_NUMERIC (1), or if it should use the C default instead (0). */
        int using_numeric_locale;
        /** Flag indicating whether this locale is actually using a locale for
         * LC_TIME (1), or if it should use the C default instead (0). */
        int using_time_locale;
        /** Flag indicating whether this locale is actually using a locale for
         * LC_MESSAGES (1), or if it should use the C default instead (0). */
        int using_messages_locale;
        /** The structure to be returned from localeconv_l() for this locale. */
        struct lconv lconv;
        /** Persistent state used by mblen() calls. */
        __mbstate_t mblen;
        /** Persistent state used by mbrlen() calls. */
        __mbstate_t mbrlen;
        /** Persistent state used by mbrtoc16() calls. */
        __mbstate_t mbrtoc16;
        /** Persistent state used by mbrtoc32() calls. */
        __mbstate_t mbrtoc32;
        /** Persistent state used by mbrtowc() calls. */
        __mbstate_t mbrtowc;
        /** Persistent state used by mbsnrtowcs() calls. */
        __mbstate_t mbsnrtowcs;
        /** Persistent state used by mbsrtowcs() calls. */
        __mbstate_t mbsrtowcs;
        /** Persistent state used by mbtowc() calls. */
        __mbstate_t mbtowc;
        /** Persistent state used by c16rtomb() calls. */
        __mbstate_t c16rtomb;
        /** Persistent state used by c32rtomb() calls. */
```

```
128         __mbstate_t c32rtomb;
129         /** Persistent state used by wcrtomb() calls. */
130         __mbstate_t wcrtomb;
131         /** Persistent state used by wcsnrtombs() calls. */
132         __mbstate_t wcsnrtombs;
133         /** Persistent state used by wcsrtombs() calls. */
134         __mbstate_t wcsrtombs;
135         /** Persistent state used by wctomb() calls. */
136         __mbstate_t wctomb;
137         /** Buffer used by nl_langinfo_l() */
138         char *csym;
139 };

141 /*
142  * XXX freebsd-head/include/xlocale/_ctype.h
143  */
144 // typedef struct _xlocale *locale_t;
145 #define LC_GLOBAL_LOCALE        ((locale_t)-1)

147 /*
148  * Increments the reference count of a reference-counted structure.
149  */
150 __attribute__((unused)) static void*
151 xlocale_retain(void *val)
152 {
153         struct xlocale_refcounted *obj = val;
154         //atomic_add_long(&(obj->retain_count), 1); /* XXX */
155         return (val);
156 }

158 /**
159  * Decrements the reference count of a reference-counted structure, freeing it
160  * if this is the last reference, calling its destructor if it has one.
161  */
162 __attribute__((unused)) static void
163 xlocale_release(void *val)
164 {
165         struct xlocale_refcounted *obj = val;
166         long count = 0; // atomic_fetchadd_long(&(obj->retain_count), -1) - 1; /
167         if (count < 0) {
168                 if (0 != obj->destructor) {
169                         obj->destructor(obj);
170                 }
171         }
172 }

174 /**
175  * Load functions.  Each takes the name of a locale and a pointer to the data
176  * to be initialised as arguments.  Two special values are allowed for the
177  */
178 extern void     *__collate_load(const char*, locale_t);
179 extern void     *__ctype_load(const char*, locale_t);
180 extern void     *__messages_load(const char*, locale_t);
181 extern void     *__monetary_load(const char*, locale_t);
182 extern void     *__numeric_load(const char*, locale_t);
183 extern void     *__time_load(const char*, locale_t);

185 extern struct _xlocale __xlocale_global_locale;
186 extern struct _xlocale __xlocale_C_locale;

188 /**
189  * Caches the rune table in TLS for fast access.
190  */
191 void __set_thread_rune_locale(locale_t loc);
192 /**
193  * Flag indicating whether a per-thread locale has been set.  If no per-thread
```



```
194  * locale has ever been set, then we always use the global locale.
195  */
196 extern int __has_thread_locale;

198 #ifndef __NO_TLS
199 /*
200  * The per-thread locale.  Avoids the need to use pthread lookup functions when
201  * getting the per-thread locale.
202  */
203 extern __thread locale_t __thread_locale;

205 /**
206  * Returns the current locale for this thread, or the global locale if none is
207  * set.  The caller does not have to free the locale.  The return value from
208  * this call is not guaranteed to remain valid after the locale changes.  As
209  * such, this should only be called within libc functions.
210  */
211 static inline locale_t __get_locale(void)
212 {

214         if (!__has_thread_locale) {
215                 return (&__xlocale_global_locale);
216         }
217         return (__thread_locale ? __thread_locale : &__xlocale_global_locale);
218 }
219 #else
220 locale_t __get_locale(void);
221 #endif

223 /**
224  * Two magic values are allowed for locale_t objects.  NULL and -1.  This
225  * function maps those to the real locales that they represent.
226  */
227 static inline locale_t get_real_locale(locale_t locale)
228 {
229         switch ((intptr_t)locale) {
230                 case 0: return (&__xlocale_C_locale);
231                 case -1: return (&__xlocale_global_locale);
232                 default: return (locale);
233         }
234 }

236 /**
237  * Replace a placeholder locale with the real global or thread-local locale_t.
238  */
239 #define FIX_LOCALE(l) (l = get_real_locale(l))

241 #endif
242 #endif /* ! codereview */
```

```
**********************************************************
   54791 Thu Jul 11 12:26:15 2013
new/usr/src/lib/libc/port/mapfile-vers
2964 need POSIX 2008 locale object support
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
#
# Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# Copyright (c) 2012 by Delphix. All rights reserved.
# Copyright (c) 2013, OmniTI Computer Consulting, Inc. All rights reserved.

#
# MAPFILE HEADER START
#
# WARNING:  STOP NOW.  DO NOT MODIFY THIS FILE.
# Object versioning must comply with the rules detailed in
#
#       usr/src/lib/README.mapfiles
#
# You should not be making modifications here until you've read the most current
# copy of that file. If you need help, contact a gatekeeper for guidance.
#
# MAPFILE HEADER END
#

$mapfile_version 2

#
# All function names added to this or any other libc mapfile
# must be placed under the 'protected:' designation.
# The 'global:' designation is used *only* for data
# items and for the members of the malloc() family.
#

#
# README README README README README README: how to update this file
#   1) each version of Solaris/OpenSolaris gets a version number.
#      (Actually since Solaris is actually a series of OpenSolaris releases
#       we'll just use OpenSolaris for this exercise.)
#       OpenSolaris 2008.11 gets 1.23
#       OpenSolaris 2009.04 gets 1.24
#       etc.
#   2) each project integration uses a unique version number.
```



```
#       PSARC/2008/123 gets 1.24.1
#       PSARC/2008/456 gets 1.24.2
#       etc.
#


# Mnemonic conditional input identifiers:
#
# - amd64, i386, sparc32, sparcv9: Correspond to ISA subdirectories used to
#       hold per-platform code. Note however that we use 'sparc32' instead of
#       'sparc'. Since '_sparc' is predefined to apply to, all sparc platforms,
#       naming the 32-bit version 'sparc' would be too likely to cause errors.
#
# -     lf64: Defined on platforms that offer the 32-bit largefile APIs
#
$if _ELF32
$add lf64
$endif
$if _sparc && _ELF32
$add sparc32
$endif
$if _sparc && _ELF64
$add sparcv9
$endif
$if _x86 && _ELF32
$add i386
$endif
$if _x86 && _ELF64
$add amd64
$endif

SYMBOL_VERSION ILLUMOS_0.5 {    # Illumos additions
    protected:
        strftime_l;
        nl_langinfo_l;
        duplocale;
        freelocale;
        newlocale;
        querylocale;
        uselocale;
} ILLUMOS_0.4;

#endif /* ! codereview */
SYMBOL_VERSION ILLUMOS_0.4 {    # Illumos additions
    protected:
        pipe2;
        dup3;
        mkostemp;
        mkostemps;

$if lf64
        mkostemp64;
        mkostemps64;
$endif
} ILLUMOS_0.3;

SYMBOL_VERSION ILLUMOS_0.3 {    # Illumos additions
    protected:
        assfail3;
} ILLUMOS_0.2;

SYMBOL_VERSION ILLUMOS_0.2 {    # Illumos additions
    protected:
        posix_spawn_pipe_np;
} ILLUMOS_0.1;
```

```
128 SYMBOL_VERSION ILLUMOS_0.1 {    # Illumos additions
129     protected:
130         timegm;
131 } SUNW_1.23;

133 SYMBOL_VERSION SUNW_1.23 {      # SunOS 5.11 (Solaris 11)
134     global:
135         _nl_domain_bindings;
136         _nl_msg_cat_cntr;

138 $if _ELF32
139         dl_iterate_phdr { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
140 $elif sparcv9
141         dl_iterate_phdr { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
142 $elif amd64
143         dl_iterate_phdr { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
144 $else
145 $error unknown platform
146 $endif

148     protected:

150 $if sparc32
151         __align_cpy_1;
152 $endif

154         addrtosymstr;
155         aio_cancel;
156         aiocancel;
157         aio_error;
158         aio_fsync;
159         aio_read;
160         aioread;
161         aio_return;
162         aio_suspend;
163         aiowait;
164         aio_waitn;
165         aio_write;
166         aiowrite;
167         asprintf;
168         assfail;
169         backtrace;
170         backtrace_symbols;
171         backtrace_symbols_fd;
172         canonicalize_file_name;
173         clearenv;
174         clock_getres;
175         clock_gettime;
176         clock_nanosleep;
177         clock_settime;
178         daemon;
179         dirfd;
180         door_bind;
181         door_call;
182         door_create;
183         door_cred;
184         door_getparam;
185         door_info;
186         door_return;
187         door_revoke;
188         door_server_create;
189         door_setparam;
190         door_ucred;
191         door_unbind;
192         door_xcreate;
193         err;
```



```
194         errx;
195         faccessat;
196         fchmodat;
197         fcloseall;
198         fdatasync;
199         ffsl;
200         ffsll;
201         fgetattr;
202         fls;
203         flsl;
204         flsll;
205         forkallx;
206         forkx;
207         fsetattr;
208         getattrat;
209         getdelim;
210         getline;
211         get_nprocs;
212         get_nprocs_conf;
213         getprogname;
214         htonl;
215         htonll;
216         htons;
217         linkat;
218         lio_listio;
219         memmem;
220         mkdirat;
221         mkdtemp;
222         mkfifoat;
223         mknodat;
224         mkstemps;
225         mmapobj;
226         mq_close;
227         mq_getattr;
228         mq_notify;
229         mq_open;
230         mq_receive;
231         mq_reltimedreceive_np;
232         mq_reltimedsend_np;
233         mq_send;
234         mq_setattr;
235         mq_timedreceive;
236         mq_timedsend;
237         mq_unlink;
238         nanosleep;
239         ntohl;
240         ntohll;
241         ntohs;
242         posix_fadvise;
243         posix_fallocate;
244         posix_madvise;
245         posix_memalign;
246         posix_spawn_file_actions_addclosefrom_np;
247         posix_spawnattr_getsigignore_np;
248         posix_spawnattr_setsigignore_np;
249         ppoll;
250         priv_basicset;
251         pthread_key_create_once_np;
252         pthread_mutexattr_getrobust;
253         pthread_mutexattr_setrobust;
254         pthread_mutex_consistent;
255         readlinkat;
256         sched_getparam;
257         sched_get_priority_max;
258         sched_get_priority_min;
259         sched_getscheduler;
```

```
260         sched_rr_get_interval;
261         sched_setparam;
262         sched_setscheduler;
263         sched_yield;
264         sem_close;
265         sem_destroy;
266         sem_getvalue;
267         sem_init;
268         sem_open;
269         sem_post;
270         sem_reltimedwait_np;
271         sem_timedwait;
272         sem_trywait;
273         sem_unlink;
274         sem_wait;
275         setattrat;
276         setprogname;
277         _sharefs;
278         shm_open;
279         shm_unlink;
280         sigqueue;
281         sigtimedwait;
282         sigwaitinfo;
283         smt_pause;
284         stpcpy;
285         stpncpy;
286         strcasestr;
287         strchrnul;
288         strndup;
289         strnlen;
290         strnstr;
291         strsep;
292         symlinkat;
293         thr_keycreate_once;
294         timer_create;
295         timer_delete;
296         timer_getoverrun;
297         timer_gettime;
298         timer_settime;
299         u8_strcmp;
300         u8_validate;
301         uconv_u16tou32;
302         uconv_u16tou8;
303         uconv_u32tou16;
304         uconv_u32tou8;
305         uconv_u8tou16;
306         uconv_u8tou32;
307         vasprintf;
308         verr;
309         verrx;
310         vforkx;
311         vwarn;
312         vwarnx;
313         warn;
314         warnx;
315         wcpcpy;
316         wcpncpy;
317         wcscasecmp;
318         wcsdup;
319         wcsncasecmp;
320         wcsnlen;

322 $if lf64
323         aio_cancel64;
324         aio_error64;
325         aio_fsync64;
```



```
326         aio_read64;
327         aioread64;
328         aio_return64;
329         aio_suspend64;
330         aio_waitn64;
331         aio_write64;
332         aiowrite64;
333         lio_listio64;
334         mkstemps64;
335         posix_fadvise64;
336         posix_fallocate64;
337 $endif
338 } SUNW_1.22.6;

340 SYMBOL_VERSION SUNW_1.22.6 {    # s10u9 - SunOS 5.10 (Solaris 10) patch addition
341     protected:
342         futimens;
343         utimensat;
344 } SUNW_1.22.5;

346 SYMBOL_VERSION SUNW_1.22.5 {    # s10u8 - SunOS 5.10 (Solaris 10) patch addition
347     protected:
348         getpagesizes2;
349 } SUNW_1.22.4;

351 SYMBOL_VERSION SUNW_1.22.4 {    # s10u7 - SunOS 5.10 (Solaris 10) patch addition
352     protected:
353         SUNW_1.22.4;
354 } SUNW_1.22.3;

356 SYMBOL_VERSION SUNW_1.22.3 {    # SunOS 5.10 (Solaris 10) patch additions
357     protected:
358         mutex_consistent;
359         u8_textprep_str;
360         uucopy;
361         uucopystr;
362 } SUNW_1.22.2;

364 SYMBOL_VERSION SUNW_1.22.2 {    # SunOS 5.10 (Solaris 10) patch additions
365     protected:
366         is_system_labeled;
367         ucred_getlabel;
368         _ucred_getlabel;
369 } SUNW_1.22.1;

371 SYMBOL_VERSION SUNW_1.22.1 {    # SunOS 5.10 (Solaris 10) patch additions
372     protected:
373         atomic_add_8;
374         atomic_add_8_nv;
375         atomic_add_char         { FLAGS = NODYNSORT };
376         atomic_add_char_nv      { FLAGS = NODYNSORT };
377         atomic_add_int          { FLAGS = NODYNSORT };
378         atomic_add_int_nv       { FLAGS = NODYNSORT };
379         atomic_add_ptr          { FLAGS = NODYNSORT };
380         atomic_add_ptr_nv       { FLAGS = NODYNSORT };
381         atomic_add_short        { FLAGS = NODYNSORT };
382         atomic_add_short_nv     { FLAGS = NODYNSORT };
383         atomic_and_16;
384         atomic_and_16_nv;
385         atomic_and_32_nv;
386         atomic_and_64;
387         atomic_and_64_nv;
388         atomic_and_8;
389         atomic_and_8_nv;
390         atomic_and_uchar        { FLAGS = NODYNSORT };
391         atomic_and_uchar_nv     { FLAGS = NODYNSORT };
```

```
392         atomic_and_uint_nv      { FLAGS = NODYNSORT };
393         atomic_and_ulong        { FLAGS = NODYNSORT };
394         atomic_and_ulong_nv     { FLAGS = NODYNSORT };
395         atomic_and_ushort       { FLAGS = NODYNSORT };
396         atomic_and_ushort_nv    { FLAGS = NODYNSORT };
397         atomic_cas_16;
398         atomic_cas_32;
399         atomic_cas_64;
400         atomic_cas_8;
401         atomic_cas_ptr          { FLAGS = NODYNSORT };
402         atomic_cas_uchar        { FLAGS = NODYNSORT };
403         atomic_cas_uint         { FLAGS = NODYNSORT };
404         atomic_cas_ulong        { FLAGS = NODYNSORT };
405         atomic_cas_ushort       { FLAGS = NODYNSORT };
406         atomic_clear_long_excl  { FLAGS = NODYNSORT };
407         atomic_dec_16;
408         atomic_dec_16_nv;
409         atomic_dec_32;
410         atomic_dec_32_nv;
411         atomic_dec_64;
412         atomic_dec_64_nv;
413         atomic_dec_8;
414         atomic_dec_8_nv;
415         atomic_dec_uchar        { FLAGS = NODYNSORT };
416         atomic_dec_uchar_nv     { FLAGS = NODYNSORT };
417         atomic_dec_uint         { FLAGS = NODYNSORT };
418         atomic_dec_uint_nv      { FLAGS = NODYNSORT };
419         atomic_dec_ulong        { FLAGS = NODYNSORT };
420         atomic_dec_ulong_nv     { FLAGS = NODYNSORT };
421         atomic_dec_ushort       { FLAGS = NODYNSORT };
422         atomic_dec_ushort_nv    { FLAGS = NODYNSORT };
423         atomic_inc_16;
424         atomic_inc_16_nv;
425         atomic_inc_32;
426         atomic_inc_32_nv;
427         atomic_inc_64;
428         atomic_inc_64_nv;
429         atomic_inc_8;
430         atomic_inc_8_nv;
431         atomic_inc_uchar        { FLAGS = NODYNSORT };
432         atomic_inc_uchar_nv     { FLAGS = NODYNSORT };
433         atomic_inc_uint         { FLAGS = NODYNSORT };
434         atomic_inc_uint_nv      { FLAGS = NODYNSORT };
435         atomic_inc_ulong        { FLAGS = NODYNSORT };
436         atomic_inc_ulong_nv     { FLAGS = NODYNSORT };
437         atomic_inc_ushort       { FLAGS = NODYNSORT };
438         atomic_inc_ushort_nv    { FLAGS = NODYNSORT };
439         atomic_or_16;
440         atomic_or_16_nv;
441         atomic_or_32_nv;
442         atomic_or_64;
443         atomic_or_64_nv;
444         atomic_or_8;
445         atomic_or_8_nv;
446         atomic_or_uchar         { FLAGS = NODYNSORT };
447         atomic_or_uchar_nv      { FLAGS = NODYNSORT };
448         atomic_or_uint_nv       { FLAGS = NODYNSORT };
449         atomic_or_ulong         { FLAGS = NODYNSORT };
450         atomic_or_ulong_nv      { FLAGS = NODYNSORT };
451         atomic_or_ushort        { FLAGS = NODYNSORT };
452         atomic_or_ushort_nv     { FLAGS = NODYNSORT };
453         atomic_set_long_excl    { FLAGS = NODYNSORT };
454         atomic_swap_16;
455         atomic_swap_32;
456         atomic_swap_64;
457         atomic_swap_8;
```



```
458         atomic_swap_ptr         { FLAGS = NODYNSORT };
459         atomic_swap_uchar       { FLAGS = NODYNSORT };
460         atomic_swap_uint        { FLAGS = NODYNSORT };
461         atomic_swap_ulong       { FLAGS = NODYNSORT };
462         atomic_swap_ushort      { FLAGS = NODYNSORT };
463         membar_consumer;
464         membar_enter;
465         membar_exit;
466         membar_producer;

468 $if _ELF32
469         enable_extended_FILE_stdio;
470 $endif

472 $if i386
473         # Note: atomic_[and,dec,inc,or]_64_nv are also defined above. Here,
474         # we add the NODYNSORT attribute to them. On this platform, they are
475         # aliases for the non-_nv versions. If that is changed, these lines
476         # should be removed.
477         atomic_and_64_nv        { FLAGS = NODYNSORT };
478         atomic_dec_64_nv        { FLAGS = NODYNSORT };
479         atomic_inc_64_nv        { FLAGS = NODYNSORT };
480         atomic_or_64_nv         { FLAGS = NODYNSORT };
481 $endif
482 $if _sparc
483         # Note: atomic_OP_WIDTH_nv symbols are also defined above. Here,
484         # we add the NODYNSORT attribute to them. On this platform, they are
485         # aliases for the non-_nv versions. If that is changed, these lines
486         # should be removed.
487         atomic_add_8_nv         { FLAGS = NODYNSORT };
488         atomic_and_8_nv         { FLAGS = NODYNSORT };
489         atomic_and_16_nv        { FLAGS = NODYNSORT };
490         atomic_and_32_nv        { FLAGS = NODYNSORT };
491         atomic_and_64_nv        { FLAGS = NODYNSORT };
492         atomic_dec_8_nv         { FLAGS = NODYNSORT };
493         atomic_dec_16_nv        { FLAGS = NODYNSORT };
494         atomic_dec_32_nv        { FLAGS = NODYNSORT };
495         atomic_dec_64_nv        { FLAGS = NODYNSORT };
496         atomic_inc_8_nv         { FLAGS = NODYNSORT };
497         atomic_inc_16_nv        { FLAGS = NODYNSORT };
498         atomic_inc_32_nv        { FLAGS = NODYNSORT };
499         atomic_inc_64_nv        { FLAGS = NODYNSORT };
500         atomic_or_8_nv          { FLAGS = NODYNSORT };
501         atomic_or_16_nv         { FLAGS = NODYNSORT };
502         atomic_or_32_nv         { FLAGS = NODYNSORT };
503         atomic_or_64_nv         { FLAGS = NODYNSORT };
504 $endif
505 } SUNW_1.22;

507 SYMBOL_VERSION SUNW_1.22 {      # SunOS 5.10 (Solaris 10)
508     global:
509 $if _ELF32
510         dladdr          { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
511         dladdr1         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
512         dlclose         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
513         dldump          { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
514         dlerror         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
515         dlinfo          { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
516         dlmopen         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
517         dlopen          { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
518         dlsym           { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
519 $elif sparcv9
520         dladdr          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
521         dladdr1         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
522         dlclose         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
523         dldump          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
```

```
524         dlerror         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
525         dlinfo          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
526         dlmopen         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
527         dlopen          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
528         dlsym           { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
529 $elif amd64
530         dladdr          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
531         dladdr1         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
532         dlamd64getunwind { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
533         dlclose         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
534         dldump          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
535         dlerror         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
536         dlinfo          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
537         dlmopen         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
538         dlopen          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
539         dlsym           { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
540 $else
541 $error unknown platform
542 $endif

544     protected:
545         alphasort;
546         _alphasort;
547         atomic_add_16;
548         atomic_add_16_nv;
549         atomic_add_32;
550         atomic_add_32_nv;
551         atomic_add_64;
552         atomic_add_64_nv;
553         atomic_add_long         { FLAGS = NODYNSORT };
554         atomic_add_long_nv      { FLAGS = NODYNSORT };
555         atomic_and_32;
556         atomic_and_uint         { FLAGS = NODYNSORT };
557         atomic_or_32;
558         atomic_or_uint          { FLAGS = NODYNSORT };
559         _Exit;
560         getisax;
561         _getisax;
562         getopt_clip;
563         _getopt_clip;
564         getopt_long;
565         _getopt_long;
566         getopt_long_only;
567         _getopt_long_only;
568         getpeerucred;
569         _getpeerucred;
570         getpflags;
571         _getpflags;
572         getppriv;
573         _getppriv;
574         getprivimplinfo;
575         _getprivimplinfo;
576         getzoneid;
577         getzoneidbyname;
578         getzonenamebyid;
579         imaxabs;
580         imaxdiv;
581         isblank;
582         iswblank;
583         port_alert;
584         port_associate;
585         port_create;
586         port_dissociate;
587         port_get;
588         port_getn;
589         port_send;
```



```
590         port_sendn;
591         posix_openpt;
592         posix_spawn;
593         posix_spawnattr_destroy;
594         posix_spawnattr_getflags;
595         posix_spawnattr_getpgroup;
596         posix_spawnattr_getschedparam;
597         posix_spawnattr_getschedpolicy;
598         posix_spawnattr_getsigdefault;
599         posix_spawnattr_getsigmask;
600         posix_spawnattr_init;
601         posix_spawnattr_setflags;
602         posix_spawnattr_setpgroup;
603         posix_spawnattr_setschedparam;
604         posix_spawnattr_setschedpolicy;
605         posix_spawnattr_setsigdefault;
606         posix_spawnattr_setsigmask;
607         posix_spawn_file_actions_addclose;
608         posix_spawn_file_actions_adddup2;
609         posix_spawn_file_actions_addopen;
610         posix_spawn_file_actions_destroy;
611         posix_spawn_file_actions_init;
612         posix_spawnp;
613         priv_addset;
614         _priv_addset;
615         priv_allocset;
616         _priv_allocset;
617         priv_copyset;
618         _priv_copyset;
619         priv_delset;
620         _priv_delset;
621         priv_emptyset;
622         _priv_emptyset;
623         priv_fillset;
624         _priv_fillset;
625         __priv_free_info;
626         priv_freeset;
627         _priv_freeset;
628         priv_getbyname;
629         _priv_getbyname;
630         __priv_getbyname;
631         priv_getbynum;
632         _priv_getbynum;
633         __priv_getbynum;
634         __priv_getdata;
635         priv_getsetbyname;
636         _priv_getsetbyname;
637         __priv_getsetbyname;
638         priv_getsetbynum;
639         _priv_getsetbynum;
640         __priv_getsetbynum;
641         priv_gettext;
642         _priv_gettext;
643         priv_ineffect;
644         _priv_ineffect;
645         priv_intersect;
646         _priv_intersect;
647         priv_inverse;
648         _priv_inverse;
649         priv_isemptyset;
650         _priv_isemptyset;
651         priv_isequalset;
652         _priv_isequalset;
653         priv_isfullset;
654         _priv_isfullset;
655         priv_ismember;
```

```
656         _priv_ismember;
657         priv_issubset;
658         _priv_issubset;
659         __priv_parse_info;
660         priv_set;
661         _priv_set;
662         priv_set_to_str;
663         _priv_set_to_str;
664         __priv_set_to_str;
665         priv_str_to_set;
666         _priv_str_to_set;
667         priv_union;
668         _priv_union;
669         pselect;
670         pthread_attr_getstack;
671         pthread_attr_setstack;
672         pthread_barrierattr_destroy;
673         pthread_barrierattr_getpshared;
674         pthread_barrierattr_init;
675         pthread_barrierattr_setpshared;
676         pthread_barrier_destroy;
677         pthread_barrier_init;
678         pthread_barrier_wait;
679         pthread_condattr_getclock;
680         pthread_condattr_setclock;
681         pthread_mutexattr_getrobust_np  { FLAGS = NODYNSORT };
682         pthread_mutexattr_setrobust_np  { FLAGS = NODYNSORT };
683         pthread_mutex_consistent_np     { FLAGS = NODYNSORT };
684         pthread_mutex_reltimedlock_np;
685         pthread_mutex_timedlock;
686         pthread_rwlock_reltimedrdlock_np;
687         pthread_rwlock_reltimedwrlock_np;
688         pthread_rwlock_timedrdlock;
689         pthread_rwlock_timedwrlock;
690         pthread_setschedprio;
691         pthread_spin_destroy;
692         pthread_spin_init;
693         pthread_spin_lock;
694         pthread_spin_trylock;
695         pthread_spin_unlock;
696         rctlblk_set_recipient_pid;
697         scandir;
698         _scandir;
699         schedctl_exit;
700         schedctl_init;
701         schedctl_lookup;
702         sema_reltimedwait;
703         sema_timedwait;
704         setenv;
705         setpflags;
706         _setpflags;
707         setppriv;
708         _setppriv;
709         strerror_r;
710         strtof;
711         strtoimax;
712         strtold;
713         strtoumax;
714         ucred_free;
715         _ucred_free;
716         ucred_get;
717         _ucred_get;
718         ucred_getegid;
719         _ucred_getegid;
720         ucred_geteuid;
721         _ucred_geteuid;
```



```
722         ucred_getgroups;
723         _ucred_getgroups;
724         ucred_getpflags;
725         _ucred_getpflags;
726         ucred_getpid;
727         _ucred_getpid;
728         ucred_getprivset;
729         _ucred_getprivset;
730         ucred_getprojid;
731         _ucred_getprojid;
732         ucred_getrgid;
733         _ucred_getrgid;
734         ucred_getruid;
735         _ucred_getruid;
736         ucred_getsgid;
737         _ucred_getsgid;
738         ucred_getsuid;
739         _ucred_getsuid;
740         ucred_getzoneid;
741         _ucred_getzoneid;
742         ucred_size;
743         _ucred_size;
744         unsetenv;
745         wcstof;
746         wcstoimax;
747         wcstold;
748         wcstoll;
749         wcstoull;
750         wcstoumax;

752 $if lf64
753         alphasort64;
754         _alphasort64;
755         pselect_large_fdset;
756         scandir64;
757         _scandir64;
758 $endif

760 $if _ELF64
761         walkcontext;
762 $endif

764 $if _sparc
765         # Note: atomic_add_[16,32,64]_nv are also defined above. Here, we add
766         # the NODYNSORT attribute to them. On this platform, they are aliases
767         # for the non-_nv versions. If that is changed, these lines should be
768         # removed.
769         atomic_add_16_nv        { FLAGS = NODYNSORT };
770         atomic_add_32_nv        { FLAGS = NODYNSORT };
771         atomic_add_64_nv        { FLAGS = NODYNSORT };
772 $endif

774 $if i386
775         # Note: atomic_add_64_nv is also defined above. Here, we add the
776         # NODYNSORT attribute to it. On this platform, it is an aliases for
777         # atomic_add_64. If that is changed, this line should be removed.
778         atomic_add_64_nv        { FLAGS = NODYNSORT };
779 $endif

781 $if amd64
782         # Exception unwind APIs required by the amd64 ABI
783         _SUNW_Unwind_DeleteException;
784         _SUNW_Unwind_ForcedUnwind;
785         _SUNW_Unwind_GetCFA;
786         _SUNW_Unwind_GetGR;
787         _SUNW_Unwind_GetIP;
```

```
788         _SUNW_Unwind_GetLanguageSpecificData;
789         _SUNW_Unwind_GetRegionStart;
790         _SUNW_Unwind_RaiseException;
791         _SUNW_Unwind_Resume;
792         _SUNW_Unwind_SetGR;
793         _SUNW_Unwind_SetIP;
794         _UA_CLEANUP_PHASE;
795         _UA_FORCE_UNWIND;
796         _UA_HANDLER_FRAME;
797         _UA_SEARCH_PHASE;
798         _Unwind_DeleteException;
799         _Unwind_ForcedUnwind;
800         _Unwind_GetCFA;
801         _Unwind_GetGR;
802         _Unwind_GetIP;
803         _Unwind_GetLanguageSpecificData;
804         _Unwind_GetRegionStart;
805         _Unwind_RaiseException;
806         _Unwind_Resume;
807         _Unwind_SetGR;
808         _Unwind_SetIP;
809 $endif
810 } SUNW_1.21.3;

812 SYMBOL_VERSION SUNW_1.21.3 {    # SunOS 5.9 (Solaris 9) patch additions
813     protected:
814         forkall;
815 } SUNW_1.21.2;

817 SYMBOL_VERSION SUNW_1.21.2 {    # SunOS 5.9 (Solaris 9) patch additions
818     protected:
819         getustack;
820         _getustack;
821         setustack;
822         _setustack;
823         stack_getbounds;
824         _stack_getbounds;
825         _stack_grow;
826         stack_inbounds;
827         _stack_inbounds;
828         stack_setbounds;
829         _stack_setbounds;
830         stack_violation;
831         _stack_violation;

833 $if _sparc
834         __makecontext_v2;
835         ___makecontext_v2;
836 $endif
837 } SUNW_1.21.1;

839 SYMBOL_VERSION SUNW_1.21.1 {    # SunOS 5.9 (Solaris 9) patch additions
840     protected:
841         crypt_gensalt;
842 } SUNW_1.21;

844 SYMBOL_VERSION SUNW_1.21 {      # SunOS 5.9 (Solaris 9)
845     protected:
846         attropen;
847         _attropen;
848         bind_textdomain_codeset;
849         closefrom;
850         _closefrom;
851         cond_reltimedwait;
852         dcngettext;
853         dngettext;
```



```
854         fchownat;
855         _fchownat;
856         fdopendir;
857         _fdopendir;
858         fdwalk;
859         _fdwalk;
860         fstatat;
861         _fstatat;
862         futimesat;
863         _futimesat;
864         getcpuid;
865         _getcpuid;
866         gethomelgroup;
867         _gethomelgroup          { FLAGS = NODYNSORT };
868         getpagesizes;
869         getrctl;
870         _getrctl;
871         issetugid;
872         _issetugid;
873         _lwp_cond_reltimedwait;
874         meminfo;
875         _meminfo;
876         ngettext;
877         openat;
878         _openat;
879         printstack;
880         priocntl;
881         priocntlset;
882         pset_getattr;
883         pset_getloadavg;
884         pset_list;
885         pset_setattr;
886         pthread_cond_reltimedwait_np;
887         rctlblk_get_enforced_value;
888         rctlblk_get_firing_time;
889         rctlblk_get_global_action;
890         rctlblk_get_global_flags;
891         rctlblk_get_local_action;
892         rctlblk_get_local_flags;
893         rctlblk_get_privilege;
894         rctlblk_get_recipient_pid;
895         rctlblk_get_value;
896         rctlblk_set_local_action;
897         rctlblk_set_local_flags;
898         rctlblk_set_privilege;
899         rctlblk_set_value;
900         rctlblk_size;
901         rctl_walk;
902         renameat;
903         setrctl;
904         _setrctl;
905         unlinkat;
906         _unlinkat;
907         vfscanf;
908         _vfscanf;
909         vfwscanf;
910         vscanf;
911         _vscanf;
912         vsscanf;
913         _vsscanf;
914         vswscanf;
915         vwscanf;

917 $if _ELF32
918         walkcontext;
919 $endif
```

```
$if lf64
        attropen64;
        _attropen64;
        fstatat64;
        _fstatat64;
        openat64;
        _openat64;
$endif
} SUNW_1.20.4;

SYMBOL_VERSION SUNW_1.20.4 {    # SunOS 5.8 (Solaris 8) patch additions
    protected:
        semtimedop;
        _semtimedop;
} SUNW_1.20.1;

SYMBOL_VERSION SUNW_1.20.1 {    # SunOS 5.8 (Solaris 8) patch additions
    protected:
        getacct;
        _getacct;
        getprojid;
        _getprojid;
        gettaskid;
        _gettaskid;
        msgids;
        _msgids;
        msgsnap;
        _msgsnap;
        putacct;
        _putacct;
        semids;
        _semids;
        settaskid;
        _settaskid;
        shmids;
        _shmids;
        wracct;
        _wracct;
} SUNW_1.20;

SYMBOL_VERSION SUNW_1.20 {      # SunOS 5.8 (Solaris 8)
    protected:
        getextmntent;
        resetmnttab;
} SUNW_1.19;

SYMBOL_VERSION SUNW_1.19 {
    protected:
        strlcat;
        strlcpy;
        umount2;
        _umount2;
} SUNW_1.18.1;

SYMBOL_VERSION SUNW_1.18.1 {
    protected:
        __fsetlocking;
} SUNW_1.18;

SYMBOL_VERSION SUNW_1.18 {      # SunOS 5.7 (Solaris 7)
    protected:
        btowc;
        __fbufsize;
        __flbf;
        _flushlbf;
```



```
        __fpending;
        __fpurge;
        __freadable;
        __freading;
        fwide;
        fwprintf;
        __fwritable;
        __fwriting;
        fwscanf;
        getloadavg;
        isaexec;
        mbrlen;
        mbrtowc;
        mbsinit;
        mbsrtowcs;
        pcsample;
        pthread_attr_getguardsize;
        pthread_attr_setguardsize;
        pthread_getconcurrency;
        pthread_mutexattr_gettype;
        pthread_mutexattr_settype;
        pthread_rwlockattr_destroy;
        pthread_rwlockattr_getpshared;
        pthread_rwlockattr_init;
        pthread_rwlockattr_setpshared;
        pthread_rwlock_destroy;
        pthread_rwlock_init;
        pthread_rwlock_rdlock;
        pthread_rwlock_tryrdlock;
        pthread_rwlock_trywrlock;
        pthread_rwlock_unlock;
        pthread_rwlock_wrlock;
        pthread_setconcurrency;
        swprintf;
        swscanf;
        __sysconf_xpg5;
        vfwprintf;
        vswprintf;
        vwprintf;
        wcrtomb;
        wcsrtombs;
        wcsstr;
        wctob;
        wmemchr;
        wmemcmp;
        wmemcpy;
        wmemmove;
        wmemset;
        wprintf;
        wscanf;

$if _ELF32
        select_large_fdset;
$endif
} SUNW_1.17;

# The empty versions SUNW_1.2 through SUNW_1.17 must be preserved because
# applications built on Solaris 2.6 Beta (when they did contain symbols)
# may depend on them.  All symbol content for SunOS 5.6 is now in SUNW_1.1

SYMBOL_VERSION SUNW_1.17 {
    protected:
        SUNW_1.17;
} SUNW_1.16;

SYMBOL_VERSION SUNW_1.16 {
```

```
1052     protected:
1053         SUNW_1.16;
1054 } SUNW_1.15;

1056 SYMBOL_VERSION SUNW_1.15 {
1057     protected:
1058         SUNW_1.15;
1059 } SUNW_1.14;

1061 SYMBOL_VERSION SUNW_1.14 {
1062     protected:
1063         SUNW_1.14;
1064 } SUNW_1.13;

1066 SYMBOL_VERSION SUNW_1.13 {
1067     protected:
1068         SUNW_1.13;
1069 } SUNW_1.12;

1071 SYMBOL_VERSION SUNW_1.12 {
1072     protected:
1073         SUNW_1.12;
1074 } SUNW_1.11;

1076 SYMBOL_VERSION SUNW_1.11 {
1077     protected:
1078         SUNW_1.11;
1079 } SUNW_1.10;

1081 SYMBOL_VERSION SUNW_1.10 {
1082     protected:
1083         SUNW_1.10;
1084 } SUNW_1.9;

1086 SYMBOL_VERSION SUNW_1.9 {
1087     protected:
1088         SUNW_1.9;
1089 } SUNW_1.8;

1091 SYMBOL_VERSION SUNW_1.8 {
1092     protected:
1093         SUNW_1.8;
1094 } SUNW_1.7;

1096 SYMBOL_VERSION SUNW_1.7 {
1097     protected:
1098         SUNW_1.7;
1099 } SUNW_1.6;

1101 SYMBOL_VERSION SUNW_1.6 {
1102     protected:
1103         SUNW_1.6;
1104 } SUNW_1.5;

1106 SYMBOL_VERSION SUNW_1.5 {
1107     protected:
1108         SUNW_1.5;
1109 } SUNW_1.4;

1111 SYMBOL_VERSION SUNW_1.4 {
1112     protected:
1113         SUNW_1.4;
1114 } SUNW_1.3;

1116 SYMBOL_VERSION SUNW_1.3 {
1117     protected:
```



```
1118         SUNW_1.3;
1119 } SUNW_1.2;

1121 SYMBOL_VERSION SUNW_1.2 {
1122     protected:
1123         SUNW_1.2;
1124 } SUNW_1.1;

1126 SYMBOL_VERSION SUNW_1.1 {       # SunOS 5.6 (Solaris 2.6)
1127     global:
1128         __loc1;
1129     protected:
1130         basename;
1131         bindtextdomain;
1132         bsd_signal;
1133         dbm_clearerr;
1134         dbm_error;
1135         dcgettext;
1136         dgettext;
1137         directio;
1138         dirname;
1139         endusershell;
1140         _exithandle;
1141         fgetwc;
1142         fgetws;
1143         fpgetround;
1144         fpsetround;
1145         fputwc;
1146         fputws;
1147         fseeko;
1148         ftello;
1149         ftrylockfile;
1150         getexecname;
1151         _getexecname;
1152         getpassphrase;
1153         gettext;
1154         getusershell;
1155         getwc;
1156         getwchar;
1157         getws;
1158         isenglish;
1159         isideogram;
1160         isnumber;
1161         isphonogram;
1162         isspecial;
1163         iswalnum;
1164         iswalpha;
1165         iswcntrl;
1166         iswctype;
1167         iswdigit;
1168         iswgraph;
1169         iswlower;
1170         iswprint;
1171         iswpunct;
1172         iswspace;
1173         iswupper;
1174         iswxdigit;
1175         ___loc1;
1176         _longjmp;
1177         _lwp_sema_trywait;
1178         ntp_adjtime;
1179         _ntp_adjtime;
1180         ntp_gettime;
1181         _ntp_gettime;
1182         __posix_asctime_r;
1183         __posix_ctime_r;
```

```
1184         __posix_getgrgid_r;
1185         __posix_getgrnam_r;
1186         __posix_getlogin_r;
1187         __posix_getpwnam_r;
1188         __posix_getpwuid_r;
1189         __posix_sigwait;
1190         __posix_ttyname_r;
1191         pset_assign;
1192         pset_bind;
1193         pset_create;
1194         pset_destroy;
1195         pset_info;
1196         pthread_atfork;
1197         pthread_attr_destroy;
1198         pthread_attr_getdetachstate;
1199         pthread_attr_getinheritsched;
1200         pthread_attr_getschedparam;
1201         pthread_attr_getschedpolicy;
1202         pthread_attr_getscope;
1203         pthread_attr_getstackaddr;
1204         pthread_attr_getstacksize;
1205         pthread_attr_init;
1206         pthread_attr_setdetachstate;
1207         pthread_attr_setinheritsched;
1208         pthread_attr_setschedparam;
1209         pthread_attr_setschedpolicy;
1210         pthread_attr_setscope;
1211         pthread_attr_setstackaddr;
1212         pthread_attr_setstacksize;
1213         pthread_cancel;
1214         __pthread_cleanup_pop;
1215         __pthread_cleanup_push;
1216         pthread_create;
1217         pthread_detach;
1218         pthread_equal;
1219         pthread_exit;
1220         pthread_getschedparam;
1221         pthread_getspecific;
1222         pthread_join;
1223         pthread_key_create;
1224         pthread_key_delete;
1225         pthread_kill;
1226         pthread_once;
1227         pthread_self;
1228         pthread_setcancelstate;
1229         pthread_setcanceltype;
1230         pthread_setschedparam;
1231         pthread_setspecific;
1232         pthread_sigmask;
1233         pthread_testcancel;
1234         putwc;
1235         putwchar;
1236         putws;
1237         regcmp;
1238         regex;
1239         resolvepath;
1240         _resolvepath;
1241         rwlock_destroy          { FLAGS = NODYNSORT };
1242         _rwlock_destroy         { FLAGS = NODYNSORT };
1243         sema_destroy;
1244         _sema_destroy;
1245         _setjmp;
1246         setusershell;
1247         siginterrupt;
1248         sigstack;
1249         snprintf;
```



```
1250         strtows;
1251         sync_instruction_memory;
1252         textdomain;
1253         thr_main;
1254         towctrans;
1255         towlower;
1256         towupper;
1257         ungetwc;
1258         vsnprintf;
1259         watoll;
1260         wcscat;
1261         wcschr;
1262         wcscmp;
1263         wcscoll;
1264         wcscpy;
1265         wcscspn;
1266         wcsftime;
1267         wcslen;
1268         wcsncat;
1269         wcsncmp;
1270         wcsncpy;
1271         wcspbrk;
1272         wcsrchr;
1273         wcsspn;
1274         wcstod;
1275         wcstok;
1276         wcstol;
1277         wcstoul;
1278         wcswcs;
1279         wcswidth;
1280         wcsxfrm;
1281         wctrans;
1282         wctype;
1283         wcwidth;
1284         wscasecmp;
1285         wscat;
1286         wschr;
1287         wscmp;
1288         wscol;
1289         wscoll;
1290         wscpy;
1291         wscspn;
1292         wsdup;
1293         wslen;
1294         wsncasecmp;
1295         wsncat;
1296         wsncmp;
1297         wsncpy;
1298         wspbrk;
1299         wsprintf;
1300         wsrchr;
1301         wsscanf;
1302         wsspn;
1303         wstod;
1304         wstok;
1305         wstol;
1306         wstoll;
1307         wstostr;
1308         wsxfrm;
1309         __xpg4_putmsg;
1310         __xpg4_putpmsg;

1312 $if lf64
1313         creat64;
1314         _creat64;
1315         fgetpos64;
```

```
1316         fopen64;
1317         freopen64;
1318         fseeko64;
1319         fsetpos64;
1320         fstat64;
1321         _fstat64;
1322         fstatvfs64;
1323         _fstatvfs64;
1324         ftello64;
1325         ftruncate64;
1326         _ftruncate64;
1327         ftw64;
1328         _ftw64;
1329         getdents64;
1330         _getdents64;
1331         getrlimit64;
1332         _getrlimit64;
1333         lockf64;
1334         _lockf64;
1335         lseek64;
1336         _lseek64;
1337         lstat64;
1338         _lstat64;
1339         mkstemp64;
1340         _mkstemp64;
1341         mmap64;
1342         _mmap64;
1343         nftw64;
1344         _nftw64;
1345         open64;
1346         _open64;
1347         __posix_readdir_r;
1348         pread64;
1349         _pread64;
1350         pwrite64;
1351         _pwrite64;
1352         readdir64;
1353         _readdir64;
1354         readdir64_r;
1355         _readdir64_r;
1356         setrlimit64;
1357         _setrlimit64;
1358         s_fcntl;
1359         _s_fcntl                { FLAGS = NODYNSORT };
1360         s_ioctl;
1361         stat64;
1362         _stat64;
1363         statvfs64;
1364         _statvfs64;
1365         tell64;
1366         _tell64;
1367         tmpfile64;
1368         truncate64;
1369         _truncate64;
1370         _xftw64;
1371 $endif

1373 $if _sparc
1374         __flt_rounds;
1375 $endif
1376 } SUNW_0.9;

1378 SYMBOL_VERSION SUNW_0.9 {       # SunOS 5.5 (Solaris 2.5)
1379     protected:
1380         acl;
1381         bcmp;
```



```
1382         bcopy;
1383         bzero;
1384         facl;
1385         ftime;
1386         getdtablesize;
1387         gethostid;
1388         gethostname;
1389         getpagesize;
1390         getpriority;
1391         getrusage;
1392         getwd;
1393         index;
1394         initstate;
1395         killpg;
1396         _nsc_trydoorcall;
1397         pthread_condattr_destroy;
1398         pthread_condattr_getpshared;
1399         pthread_condattr_init;
1400         pthread_condattr_setpshared;
1401         pthread_cond_broadcast;
1402         pthread_cond_destroy;
1403         pthread_cond_init;
1404         pthread_cond_signal;
1405         pthread_cond_timedwait;
1406         pthread_cond_wait;
1407         pthread_mutexattr_destroy;
1408         pthread_mutexattr_getprioceiling;
1409         pthread_mutexattr_getprotocol;
1410         pthread_mutexattr_getpshared;
1411         pthread_mutexattr_init;
1412         pthread_mutexattr_setprioceiling;
1413         pthread_mutexattr_setprotocol;
1414         pthread_mutexattr_setpshared;
1415         pthread_mutex_destroy;
1416         pthread_mutex_getprioceiling;
1417         pthread_mutex_init;
1418         pthread_mutex_lock;
1419         pthread_mutex_setprioceiling;
1420         pthread_mutex_trylock;
1421         pthread_mutex_unlock;
1422         random;
1423         reboot;
1424         re_comp;
1425         re_exec;
1426         rindex;
1427         setbuffer;
1428         sethostname;
1429         setlinebuf;
1430         setpriority;
1431         setregid;
1432         setreuid;
1433         setstate;
1434         srandom;
1435         thr_min_stack;
1436         thr_stksegment;
1437         ualarm;
1438         usleep;
1439         wait3;
1440         wait4;
1441 } SUNW_0.8;

1443 SYMBOL_VERSION SUNW_0.8 {       # SunOS 5.4 (Solaris 2.4)
1444     global:
1445         __xpg4                  { FLAGS = NODIRECT };
1446     protected:
1447         addsev;
```

```
1448         cond_broadcast          { FLAGS = NODYNSORT };
1449         cond_destroy            { FLAGS = NODYNSORT };
1450         cond_init;
1451         cond_signal             { FLAGS = NODYNSORT };
1452         cond_timedwait;
1453         cond_wait;
1454         confstr;
1455         fnmatch;
1456         _getdate_err_addr;
1457         glob;
1458         globfree;
1459         iconv;
1460         iconv_close;
1461         iconv_open;
1462         lfmt;
1463         mutex_destroy           { FLAGS = NODYNSORT };
1464         mutex_init;
1465         mutex_lock              { FLAGS = NODYNSORT };
1466         mutex_trylock           { FLAGS = NODYNSORT };
1467         mutex_unlock            { FLAGS = NODYNSORT };
1468         pfmt;
1469         regcomp;
1470         regerror;
1471         regexec;
1472         regfree;
1473         rwlock_init;
1474         rw_rdlock               { FLAGS = NODYNSORT };
1475         rw_read_held;
1476         rw_tryrdlock            { FLAGS = NODYNSORT };
1477         rw_trywrlock            { FLAGS = NODYNSORT };
1478         rw_unlock               { FLAGS = NODYNSORT };
1479         rw_write_held;
1480         rw_wrlock               { FLAGS = NODYNSORT };
1481         sema_held;
1482         sema_init;
1483         sema_post;
1484         sema_trywait;
1485         sema_wait;
1486         setcat;
1487         sigfpe;
1488         strfmon;
1489         strptime;
1490         thr_continue;
1491         thr_create;
1492         thr_exit;
1493         thr_getconcurrency;
1494         thr_getprio;
1495         thr_getspecific;
1496         thr_join;
1497         thr_keycreate;
1498         thr_kill;
1499         thr_self                { FLAGS = NODYNSORT };
1500         thr_setconcurrency;
1501         thr_setprio;
1502         thr_setspecific;
1503         thr_sigsetmask;
1504         thr_suspend;
1505         thr_yield;
1506         vlfmt;
1507         vpfmt;
1508         wordexp;
1509         wordfree;
1510 } SUNW_0.7;

1512 SYMBOL_VERSION SUNW_0.7 {       # SunOS 5.3 (Solaris 2.3)
1513     global:
```



```
1514         altzone;
1515         _ctype;
1516         isnanf                  { TYPE = FUNCTION; FILTER = libm.so.2 };
1517         lone;
1518         lten;
1519         lzero;
1520         memalign                { FLAGS = NODIRECT };
1521         modff                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1522         nss_default_finders;
1523         _sibuf;
1524         _sobuf;
1525         _sys_buslist;
1526         _sys_cldlist;
1527         _sys_fpelist;
1528         _sys_illlist;
1529         _sys_segvlist;
1530         _sys_siginfolistp;
1531         _sys_siglist;
1532         _sys_siglistn;
1533         _sys_siglistp;
1534         _sys_traplist;
1535         valloc                  { FLAGS = NODIRECT };

1537 $if _ELF32
1538         _bufendtab;
1539         _lastbuf;
1540         sys_errlist;
1541         sys_nerr;
1542         _sys_nsig;
1543 $endif

1545     protected:
1546         a64l;
1547         adjtime;
1548         ascftime;
1549         _assert;
1550         atoll;
1551         brk;
1552         __builtin_alloca;
1553         cftime;
1554         closelog;
1555         csetcol;
1556         csetlen;
1557         ctermid_r;
1558         dbm_close;
1559         dbm_delete;
1560         dbm_fetch;
1561         dbm_firstkey;
1562         dbm_nextkey;
1563         dbm_open;
1564         dbm_store;
1565         decimal_to_double;
1566         decimal_to_extended;
1567         decimal_to_quadruple;
1568         decimal_to_single;
1569         double_to_decimal;
1570         drand48;
1571         econvert;
1572         ecvt;
1573         endnetgrent;
1574         endspent;
1575         endutent;
1576         endutxent;
1577         erand48;
1578         euccol;
1579         euclen;
```

```
1580        eucscol;
1581        extended_to_decimal;
1582        fchroot;
1583        fconvert;
1584        fcvt;
1585        ffs;
1586        fgetspent;
1587        fgetspent_r;
1588        _filbuf;
1589        file_to_decimal;
1590        finite;
1591        _flsbuf;
1592        fork1                   { FLAGS = NODYNSORT };
1593        fpclass;
1594        fpgetmask;
1595        fpgetsticky;
1596        fpsetmask;
1597        fpsetsticky;
1598        fstatfs;
1599        ftruncate;
1600        ftw;
1601        func_to_decimal;
1602        gconvert;
1603        gcvt;
1604        getdents;
1605        gethrtime;
1606        gethrvtime;
1607        getmntany;
1608        getmntent;
1609        getnetgrent;
1610        getnetgrent_r;
1611        getpw;
1612        getspent;
1613        getspent_r;
1614        getspnam;
1615        getspnam_r;
1616        getutent;
1617        getutid;
1618        getutline;
1619        getutmp;
1620        getutmpx;
1621        getutxent;
1622        getutxid;
1623        getutxline;
1624        getvfsany;
1625        getvfsent;
1626        getvfsfile;
1627        getvfsspec;
1628        getwidth;
1629        gsignal;
1630        hasmntopt;
1631        innetgr;
1632        insque;
1633        _insque;
1634        jrand48;
1635        l64a;
1636        ladd;
1637        lckpwdf;
1638        lcong48;
1639        ldivide;
1640        lexp10;
1641        llabs;
1642        lldiv;
1643        llog10;
1644        llseek;
1645        lltostr;
```



```
1646        lmul;
1647        lrand48;
1648        lshiftl;
1649        lsub;
1650        _lwp_cond_broadcast;
1651        _lwp_cond_signal;
1652        _lwp_cond_timedwait;
1653        _lwp_cond_wait;
1654        _lwp_continue;
1655        _lwp_info;
1656        _lwp_kill;
1657        _lwp_mutex_lock;
1658        _lwp_mutex_trylock;
1659        _lwp_mutex_unlock;
1660        _lwp_self;
1661        _lwp_sema_init;
1662        _lwp_sema_post;
1663        _lwp_sema_wait;
1664        _lwp_suspend;
1665        madvise;
1666        __major;
1667        __makedev;
1668        mincore;
1669        __minor;
1670        mkstemp;
1671        _mkstemp;
1672        mlockall;
1673        mrand48;
1674        munlockall;
1675        _mutex_held             { FLAGS = NODYNSORT };
1676        _mutex_lock             { FLAGS = NODYNSORT };
1677        nrand48;
1678        _nss_netdb_aliases;
1679        _nss_XbyY_buf_alloc;
1680        _nss_XbyY_buf_free;
1681        __nsw_extended_action;
1682        __nsw_freeconfig;
1683        __nsw_getconfig;
1684        openlog;
1685        plock;
1686        p_online;
1687        pread;
1688        __priocntl;
1689        __priocntlset;
1690        processor_bind;
1691        processor_info;
1692        psiginfo;
1693        psignal;
1694        putpwent;
1695        putspent;
1696        pututline;
1697        pututxline;
1698        pwrite;
1699        qeconvert;
1700        qecvt;
1701        qfconvert;
1702        qfcvt;
1703        qgconvert;
1704        qgcvt;
1705        quadruple_to_decimal;
1706        realpath;
1707        remque;
1708        _remque;
1709        _rw_read_held;
1710        _rw_write_held;
1711        seconvert;
```

```
1712         seed48;
1713         select;
1714         _sema_held;
1715         setegid;
1716         seteuid;
1717         setlogmask;
1718         setnetgrent;
1719         setspent;
1720         settimeofday;
1721         setutent;
1722         setutxent;
1723         sfconvert;
1724         sgconvert;
1725         sig2str;
1726         sigwait;
1727         single_to_decimal;
1728         srand48;
1729         ssignal;
1730         statfs;
1731         str2sig;
1732         strcasecmp;
1733         string_to_decimal;
1734         strncasecmp;
1735         strsignal;
1736         strtoll;
1737         strtoull;
1738         swapctl;
1739         _syscall;
1740         sysfs;
1741         syslog;
1742         _syslog;
1743         tmpnam_r;
1744         truncate;
1745         ttyslot;
1746         uadmin;
1747         ulckpwdf;
1748         ulltostr;
1749         unordered;
1750         updwtmp;
1751         updwtmpx;
1752         ustat;
1753         utimes;
1754         utmpname;
1755         utmpxname;
1756         vfork;
1757         vhangup;
1758         vsyslog;
1759         yield;

1761 $if i386
1762         # Note: _syscall is also defined above. Here, we add the NODYNSORT
1763         # attribute to it. On this platform, it is an alias to syscall.
1764         # If that is changed, this lines should be removed.
1765         _syscall                { FLAGS = NODYNSORT };
1766 $endif

1768 # The 32-bit sparc ABI requires SISCD_2.3. On other platforms, those symbols
1769 # go directly into SUNW_0.7.
1770 $if sparc32
1771 } SISCD_2.3;

1773 SYMBOL_VERSION SISCD_2.3 {
1774 $endif

1776     global:
1777         errno                   { FLAGS = NODIRECT };
```



```
1778         _iob;

1780     protected:
1781         addseverity;
1782         _addseverity;
1783         asctime_r;
1784         crypt;
1785         _crypt;
1786         ctime_r;
1787         encrypt;
1788         _encrypt;
1789         endgrent;
1790         endpwent;
1791         ___errno;
1792         fgetgrent;
1793         fgetgrent_r;
1794         fgetpwent;
1795         fgetpwent_r;
1796         flockfile;
1797         funlockfile;
1798         getchar_unlocked;
1799         getc_unlocked;
1800         getgrent;
1801         getgrent_r;
1802         getgrgid_r;
1803         getgrnam_r;
1804         getitimer;
1805         _getitimer;
1806         getlogin_r;
1807         getpwent;
1808         getpwent_r;
1809         getpwnam_r;
1810         getpwuid_r;
1811         gettimeofday;
1812         _gettimeofday;
1813         gmtime_r;
1814         localtime_r;
1815         putchar_unlocked;
1816         putc_unlocked;
1817         rand_r;
1818         readdir_r;
1819         setgrent;
1820         setitimer;
1821         _setitimer;
1822         setkey;
1823         _setkey;
1824         setpwent;
1825         strtok_r;
1826         sysinfo;
1827         _sysinfo;
1828         ttyname_r;

1830 $if _ELF32
1831         __div64;
1832         __mul64;
1833         __rem64;
1834         __udiv64;
1835         __urem64;
1836 $endif

1838 $if sparc32
1839         __dtoll;
1840         __dtoull;
1841         __ftoll;
1842         __ftoull;
1843         _Q_lltoq;
```

```
1844         _Q_qtoll;
1845         _Q_qtoull;
1846         _Q_ulltoq;
1847         sbrk;
1848         _sbrk;
1849         __umul64                { FLAGS = NODYNSORT };  # Same address as __mul6
1850 $endif

1852 # On 32-bit platforms, the following symbols go into SYSVABI_1.3, but on
1853 # other platforms they go directly into the current version (which will be
1854 # either SUNW_0.7, or SISCD_2.3, depending on the similar issue described above.
1855 $if _ELF32
1856 } SYSVABI_1.3;

1858 SYMBOL_VERSION SYSVABI_1.3 {
1859 $endif

1861     global:
1862         _altzone;
1863         calloc                  { FLAGS = NODIRECT };
1864         __ctype;
1865         daylight;
1866         _daylight;
1867         environ                 { FLAGS = NODIRECT };
1868         _environ                { FLAGS = NODIRECT };
1869         free                    { FLAGS = NODIRECT };
1870         frexp                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1871         getdate_err;
1872         _getdate_err;
1873         getenv;
1874         __huge_val;
1875         __iob;
1876         isnan                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1877         _isnan                  { TYPE = FUNCTION; FILTER = libm.so.2 };
1878         isnand                  { TYPE = FUNCTION; FILTER = libm.so.2 };
1879         _isnand                 { TYPE = FUNCTION; FILTER = libm.so.2 };
1880         ldexp                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1881         logb                    { TYPE = FUNCTION; FILTER = libm.so.2 };
1882         malloc                  { FLAGS = NODIRECT };
1883         memcmp;
1884         memcpy;
1885         memmove;
1886         memset;
1887         modf                    { TYPE = FUNCTION; FILTER = libm.so.2 };
1888         _modf                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1889         nextafter               { TYPE = FUNCTION; FILTER = libm.so.2 };
1890         _nextafter              { TYPE = FUNCTION; FILTER = libm.so.2 };
1891         _numeric;
1892         optarg;
1893         opterr;
1894         optind;
1895         optopt;
1896         realloc                 { FLAGS = NODIRECT };
1897         scalb                   { TYPE = FUNCTION; FILTER = libm.so.2 };
1898         _scalb                  { TYPE = FUNCTION; FILTER = libm.so.2 };
1899         timezone;
1900         _timezone;
1901         tzname;
1902         _tzname;
1903 $if i386
1904         _fp_hw;
1905 $endif

1907     protected:
1908         abort;
1909         abs;
```



```
1910         access;
1911         _access;
1912         acct;
1913         _acct;
1914         alarm;
1915         _alarm;
1916         asctime;
1917         __assert;
1918         atexit;
1919         atof;
1920         atoi;
1921         atol;
1922         bsearch;
1923         catclose;
1924         _catclose;
1925         catgets;
1926         _catgets;
1927         catopen;
1928         _catopen;
1929         cfgetispeed;
1930         _cfgetispeed;
1931         cfgetospeed;
1932         _cfgetospeed;
1933         cfsetispeed;
1934         _cfsetispeed;
1935         cfsetospeed;
1936         _cfsetospeed;
1937         chdir;
1938         _chdir;
1939         chmod;
1940         _chmod;
1941         chown;
1942         _chown;
1943         chroot;
1944         _chroot;
1945         _cleanup;
1946         clearerr;
1947         clock;
1948         _close;
1949         close;
1950         closedir;
1951         _closedir;
1952         creat;
1953         _creat;
1954         ctermid;
1955         ctime;
1956         cuserid;
1957         _cuserid;
1958         difftime;
1959         div;
1960         dup;
1961         _dup;
1962         dup2;
1963         _dup2;
1964         execl;
1965         _execl;
1966         execle;
1967         _execle;
1968         execlp;
1969         _execlp;
1970         execv;
1971         _execv;
1972         execve;
1973         _execve;
1974         execvp;
1975         _execvp;
```

```
1976        exit;
1977        _exit;
1978        fattach;
1979        _fattach;
1980        fchdir;
1981        _fchdir;
1982        fchmod;
1983        _fchmod;
1984        fchown;
1985        _fchown;
1986        fclose;
1987        fcntl;
1988        _fcntl;
1989        fdetach;
1990        _fdetach;
1991        fdopen;
1992        _fdopen;
1993        feof;
1994        ferror;
1995        fflush;
1996        fgetc;
1997        fgetpos;
1998        fgets;
1999        __filbuf;
2000        fileno;
2001        _fileno;
2002        __flsbuf;
2003        fmtmsg;
2004        _fmtmsg;
2005        fopen;
2006        _fork;
2007        fork;
2008        fpathconf;
2009        _fpathconf;
2010        fprintf;
2011        fputc;
2012        fputs;
2013        fread;
2014        freopen;
2015        fscanf;
2016        fseek;
2017        fsetpos;
2018        fstat;
2019        _fstat;
2020        fstatvfs;
2021        _fstatvfs;
2022        fsync;
2023        _fsync;
2024        ftell;
2025        ftok;
2026        _ftok;
2027        fwrite;
2028        getc;
2029        getchar;
2030        getcontext;
2031        _getcontext;
2032        getcwd;
2033        _getcwd;
2034        getdate;
2035        _getdate;
2036        getegid;
2037        _getegid;
2038        geteuid;
2039        _geteuid;
2040        getgid;
2041        _getgid;
```



```
2042        getgrgid;
2043        getgrnam;
2044        getgroups;
2045        _getgroups;
2046        getlogin;
2047        getmsg;
2048        _getmsg;
2049        getopt;
2050        _getopt;
2051        getpass;
2052        _getpass;
2053        getpgid;
2054        _getpgid;
2055        getpgrp;
2056        _getpgrp;
2057        getpid;
2058        _getpid;
2059        getpmsg;
2060        _getpmsg;
2061        getppid;
2062        _getppid;
2063        getpwnam;
2064        getpwuid;
2065        getrlimit;
2066        _getrlimit;
2067        gets;
2068        getsid;
2069        _getsid;
2070        getsubopt;
2071        _getsubopt;
2072        gettxt;
2073        _gettxt;
2074        getuid;
2075        _getuid;
2076        getw;
2077        _getw;
2078        gmtime;
2079        grantpt;
2080        _grantpt;
2081        hcreate;
2082        _hcreate;
2083        hdestroy;
2084        _hdestroy;
2085        hsearch;
2086        _hsearch;
2087        initgroups;
2088        _initgroups;
2089        ioctl;
2090        _ioctl;
2091        isalnum;
2092        isalpha;
2093        isascii;
2094        _isascii;
2095        isastream;
2096        _isastream;
2097        isatty;
2098        _isatty;
2099        iscntrl;
2100        isdigit;
2101        isgraph;
2102        islower;
2103        isprint;
2104        ispunct;
2105        isspace;
2106        isupper;
2107        isxdigit;
```

```
2108         kill;
2109         _kill;
2110         labs;
2111         lchown;
2112         _lchown;
2113         ldiv;
2114         lfind;
2115         _lfind;
2116         link;
2117         _link;
2118         localeconv;
2119         localtime;
2120         lockf;
2121         _lockf;
2122         longjmp;
2123         lsearch;
2124         _lsearch;
2125         lseek;
2126         _lseek;
2127         lstat;
2128         _lstat;
2129         makecontext;
2130         _makecontext;
2131         mblen;
2132         mbstowcs;
2133         mbtowc;
2134         memccpy;
2135         _memccpy;
2136         memchr;
2137         memcntl;
2138         _memcntl;
2139         mkdir;
2140         _mkdir;
2141         mkfifo;
2142         _mkfifo;
2143         mknod;
2144         _mknod;
2145         mktemp;
2146         _mktemp;
2147         mktime;
2148         mlock;
2149         _mlock;
2150         mmap;
2151         _mmap;
2152         monitor;
2153         _monitor;
2154         mount;
2155         _mount;
2156         mprotect;
2157         _mprotect;
2158         msgctl;
2159         _msgctl;
2160         msgget;
2161         _msgget;
2162         msgrcv;
2163         _msgrcv;
2164         msgsnd;
2165         _msgsnd;
2166         msync;
2167         _msync;
2168         munlock;
2169         _munlock;
2170         munmap;
2171         _munmap;
2172         nftw;
2173         _nftw;
```



```
2174         nice;
2175         _nice;
2176         nl_langinfo;
2177         _nl_langinfo;
2178         open;
2179         _open;
2180         opendir;
2181         _opendir;
2182         pathconf;
2183         _pathconf;
2184         pause;
2185         _pause;
2186         pclose;
2187         _pclose;
2188         perror;
2189         pipe;
2190         _pipe;
2191         poll;
2192         _poll;
2193         popen;
2194         _popen;
2195         printf;
2196         profil;
2197         _profil;
2198         ptsname;
2199         _ptsname;
2200         putc;
2201         putchar;
2202         putenv;
2203         _putenv;
2204         putmsg;
2205         _putmsg;
2206         putpmsg;
2207         _putpmsg;
2208         puts;
2209         putw;
2210         _putw;
2211         qsort;
2212         raise;
2213         rand;
2214         read;
2215         _read;
2216         readdir;
2217         _readdir;
2218         readlink;
2219         _readlink;
2220         readv;
2221         _readv;
2222         remove;
2223         rename;
2224         _rename;
2225         rewind;
2226         rewinddir;
2227         _rewinddir;
2228         rmdir;
2229         _rmdir;
2230         scanf;
2231         seekdir;
2232         _seekdir;
2233         semctl;
2234         _semctl;
2235         semget;
2236         _semget;
2237         semop;
2238         _semop;
2239         setbuf;
```

```
2240        setcontext;
2241        _setcontext             { FLAGS = NODYNSORT };
2242        setgid;
2243        _setgid;
2244        setgroups;
2245        _setgroups;
2246        setjmp;
2247        setlabel;
2248        setlocale;
2249        setpgid;
2250        _setpgid;
2251        setpgrp;
2252        _setpgrp;
2253        setrlimit;
2254        _setrlimit;
2255        setsid;
2256        _setsid;
2257        setuid;
2258        _setuid;
2259        setvbuf;
2260        shmat;
2261        _shmat;
2262        shmctl;
2263        _shmctl;
2264        shmdt;
2265        _shmdt;
2266        shmget;
2267        _shmget;
2268        sigaction;
2269        _sigaction              { FLAGS = NODYNSORT };
2270        sigaddset;
2271        _sigaddset;
2272        sigaltstack;
2273        _sigaltstack;
2274        sigdelset;
2275        _sigdelset;
2276        sigemptyset;
2277        _sigemptyset;
2278        sigfillset;
2279        _sigfillset;
2280        sighold;
2281        _sighold;
2282        sigignore;
2283        _sigignore;
2284        sigismember;
2285        _sigismember;
2286        siglongjmp;
2287        _siglongjmp;
2288        signal;
2289        sigpause;
2290        _sigpause;
2291        sigpending;
2292        _sigpending;
2293        sigprocmask;
2294        _sigprocmask;
2295        sigrelse;
2296        _sigrelse;
2297        sigsend;
2298        _sigsend;
2299        sigsendset;
2300        _sigsendset;
2301        sigset;
2302        _sigset;
2303        sigsetjmp;
2304        _sigsetjmp              { FLAGS = NODYNSORT };
2305        sigsuspend;
```



```
2306        _sigsuspend;
2307        sleep;
2308        _sleep;
2309        sprintf;
2310        srand;
2311        sscanf;
2312        stat;
2313        _stat;
2314        statvfs;
2315        _statvfs;
2316        stime;
2317        _stime;
2318        strcat;
2319        strchr;
2320        strcmp;
2321        strcoll;
2322        strcpy;
2323        strcspn;
2324        strdup;
2325        _strdup;
2326        strerror;
2327        strftime;
2328        strlen;
2329        strncat;
2330        strncmp;
2331        strncpy;
2332        strpbrk;
2333        strrchr;
2334        strspn;
2335        strstr;
2336        strtod;
2337        strtok;
2338        strtol;
2339        strtoul;
2340        strxfrm;
2341        swab;
2342        _swab;
2343        swapcontext;
2344        _swapcontext;
2345        symlink;
2346        _symlink;
2347        sync;
2348        _sync;
2349        sysconf;
2350        _sysconf;
2351        system;
2352        tcdrain;
2353        _tcdrain;
2354        tcflow;
2355        _tcflow;
2356        tcflush;
2357        _tcflush;
2358        tcgetattr;
2359        _tcgetattr;
2360        tcgetpgrp;
2361        _tcgetpgrp;
2362        tcgetsid;
2363        _tcgetsid;
2364        tcsendbreak;
2365        _tcsendbreak;
2366        tcsetattr;
2367        _tcsetattr;
2368        tcsetpgrp;
2369        _tcsetpgrp;
2370        tdelete;
2371        _tdelete;
```

```
2372         tell;
2373         _tell;
2374         telldir;
2375         _telldir;
2376         tempnam;
2377         _tempnam;
2378         tfind;
2379         _tfind;
2380         time;
2381         _time;
2382         times;
2383         _times;
2384         tmpfile;
2385         tmpnam;
2386         toascii;
2387         _toascii;
2388         tolower;
2389         _tolower;
2390         toupper;
2391         _toupper;
2392         tsearch;
2393         _tsearch;
2394         ttyname;
2395         twalk;
2396         _twalk;
2397         tzset;
2398         _tzset;
2399         ulimit;
2400         _ulimit;
2401         umask;
2402         _umask;
2403         umount;
2404         _umount;
2405         uname;
2406         _uname;
2407         ungetc;
2408         unlink;
2409         _unlink;
2410         unlockpt;
2411         _unlockpt;
2412         utime;
2413         _utime;
2414         vfprintf;
2415         vprintf;
2416         vsprintf;
2417         wait;
2418         _wait;
2419         waitid;
2420         _waitid;
2421         waitpid;
2422         _waitpid;
2423         wcstombs;
2424         wctomb;
2425         write;
2426         _write;
2427         writev;
2428         _writev;
2429         _xftw;

2431 $if _ELF32
2432         ptrace;
2433         _ptrace;
2434 $endif

2436 $if i386
2437         _fxstat;
```



```
2438         _lxstat;
2439         nuname;
2440         _nuname;
2441         _xmknod;
2442         _xstat;
2443 $endif

2445 $if !sparc32
2446         sbrk;
2447 $endif

2449 $if _sparc
2450         __dtou;
2451         __ftou;
2452 $endif

2454 $if sparc32
2455         .div;
2456         .mul;
2457         .rem;
2458         .stret1;
2459         .stret2;
2460         .stret4;
2461         # .stret4 and .stret8 are the same thing
2462         .stret8                 { FLAGS = NODYNSORT };
2463         .udiv;
2464         .umul;
2465         .urem;
2466         _Q_add;
2467         _Q_cmp;
2468         _Q_cmpe;
2469         _Q_div;
2470         _Q_dtoq;
2471         _Q_feq;
2472         _Q_fge;
2473         _Q_fgt;
2474         _Q_fle;
2475         _Q_flt;
2476         _Q_fne;
2477         _Q_itoq;
2478         _Q_mul;
2479         _Q_neg;
2480         _Q_qtod;
2481         _Q_qtoi;
2482         _Q_qtos;
2483         _Q_qtou;
2484         _Q_sqrt;
2485         _Q_stoq;
2486         _Q_sub;
2487         _Q_utoq;
2488 $endif

2490 $if sparcv9
2491         # __align_cpy_1 is an alias for memcpy. Filter it out of
2492         # the .SUNW_dynsymsort section
2493         __align_cpy_1           { FLAGS = NODYNSORT };
2494         __align_cpy_16;
2495         __align_cpy_2;
2496         __align_cpy_4;
2497         # __align_cpy_8 is same as __align_cpy_16
2498         __align_cpy_8           { FLAGS = NODYNSORT };
2499         __dtoul;
2500         __ftoul;
2501         _Qp_add;
2502         _Qp_cmp;
2503         _Qp_cmpe;
```

```
2504         _Qp_div;
2505         _Qp_dtoq;
2506         _Qp_feq;
2507         _Qp_fge;
2508         _Qp_fgt;
2509         _Qp_fle;
2510         _Qp_flt;
2511         _Qp_fne;
2512         _Qp_itoq;
2513         _Qp_mul;
2514         _Qp_neg;
2515         _Qp_qtod;
2516         _Qp_qtoi;
2517         _Qp_qtos;
2518         _Qp_qtoui;
2519         _Qp_qtoux;
2520         _Qp_qtox;
2521         _Qp_sqrt;
2522         _Qp_stoq;
2523         _Qp_sub;
2524         _Qp_uitoq;
2525         _Qp_uxtoq;
2526         _Qp_xtoq;
2527         __sparc_utrap_install;
2528 $endif

2530 # On amd64, we also have SYSVABI_1.3, but it contains a small subset of
2531 # the symbols put in that version on other platforms.
2532 $if amd64
2533 } SYSVABI_1.3;

2535 SYMBOL_VERSION SYSVABI_1.3 {
2536 $endif
2537     global:
2538 $if !_sparc
2539         __flt_rounds;
2540 $endif

2542     protected:
2543         _ctermid;
2544         _getgrgid;
2545         _getgrnam;
2546         _getlogin;
2547         _getpwnam;
2548         _getpwuid;
2549         _ttyname;

2551 $if !sparc32
2552         _sbrk;
2553 $endif

2555 $if _x86
2556         _fpstart;
2557         __fpstart;
2558 $endif
2559 };




2563 # There should never be more than one SUNWprivate version.
2564 # Don't add any more.  Add new private symbols to SUNWprivate_1.1

2566 SYMBOL_VERSION SUNWprivate_1.1 {
2567     global:
2568         ___Argv                 { FLAGS = NODIRECT };
2569         cfree                   { FLAGS = NODIRECT };
```



```
2570         _cswidth;
2571         __ctype_mask;
2572         __environ_lock          { FLAGS = NODIRECT };
2573         __inf_read;
2574         __inf_written;
2575         __i_size;
2576         _isnanf                 { TYPE = FUNCTION; FILTER = libm.so.2 };
2577         __iswrune;
2578         __libc_threaded;
2579         _lib_version            { FLAGS = NODIRECT };
2580         _logb                   { TYPE = FUNCTION; FILTER = libm.so.2 };
2581         _lone                   { FLAGS = NODYNSORT };
2582         _lten                   { FLAGS = NODYNSORT };
2583         _lzero                  { FLAGS = NODYNSORT };
2584         __malloc_lock;
2585         _memcmp;
2586         _memcpy                 { FLAGS = NODYNSORT };
2587         _memmove;
2588         _memset;
2589         _modff                  { TYPE = FUNCTION; FILTER = libm.so.2 };
2590         __nan_read;
2591         __nan_written;
2592         __nextwctype;
2593         __nis_debug_bind;
2594         __nis_debug_calls;
2595         __nis_debug_file;
2596         __nis_debug_rpc;
2597         __nis_prefsrv;
2598         __nis_preftype;
2599         __nis_server;
2600         _nss_default_finders;
2601         __progname              { FLAGS = NODIRECT };
2602         _smbuf;
2603         _sp;
2604         __strdupa_str           { FLAGS = NODIRECT };
2605         __strdupa_len           { FLAGS = NODIRECT };
2606         _tdb_bootstrap;
2607         __threaded;
2608         thr_probe_getfunc_addr;
2609         __trans_lower;
2610         __trans_upper;
2611         _uberdata;
2612         __xpg6                  { FLAGS = NODIRECT };

2614 $if _ELF32
2615         _dladdr                 { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2616         _dladdr1                { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2617         _dlclose                { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2618         _dldump                 { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2619         _dlerror                { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2620         _dlinfo                 { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2621         _dlmopen                { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2622         _dlopen                 { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2623         _dlsym                  { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2624         _ld_libc                { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2625         _sys_errlist;
2626         _sys_errs;
2627         _sys_index;
2628         _sys_nerr               { FLAGS = NODYNSORT };
2629         _sys_num_err;
2630 $elif sparcv9
2631         _dladdr         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2632         _dladdr1        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2633         _dlclose        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2634         _dldump         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2635         _dlerror        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
```

```
2636         _dlinfo         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2637         _dlmopen        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2638         _dlopen         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2639         _dlsym          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2640         _ld_libc        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2641 $elif amd64
2642         _dladdr         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2643         _dladdr1        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2644         _dlamd64getunwind { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2645         _dlclose        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2646         _dldump         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2647         _dlerror        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2648         _dlinfo         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2649         _dlmopen        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2650         _dlopen         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2651         _dlsym          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2652         _ld_libc        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2653 $else
2654 $error unknown platform
2655 $endif

2657 $if _sparc
2658         __lyday_to_month;
2659         __mon_lengths;
2660         __yday_to_month;
2661 $endif
2662 $if i386
2663         _sse_hw;
2664 $endif

2666     protected:
2667         acctctl;
2668         allocids;
2669         _assert_c99;
2670         __assert_c99;
2671         _assfail;
2672         attr_count;
2673         attr_to_data_type;
2674         attr_to_name;
2675         attr_to_option;
2676         attr_to_xattr_view;
2677         _autofssys;
2678         _bufsync;
2679         _cladm;
2680         __class_quadruple;
2681         core_get_default_content;
2682         core_get_default_path;
2683         core_get_global_content;
2684         core_get_global_path;
2685         core_get_options;
2686         core_get_process_content;
2687         core_get_process_path;
2688         core_set_default_content;
2689         core_set_default_path;
2690         core_set_global_content;
2691         core_set_global_path;
2692         core_set_options;
2693         core_set_process_content;
2694         core_set_process_path;
2695         dbm_close_status;
2696         dbm_do_nextkey;
2697         dbm_setdefwrite;
2698         _D_cplx_div;
2699         _D_cplx_div_ix;
2700         _D_cplx_div_rx;
2701         _D_cplx_mul;
```



```
2702         defclose_r;
2703         defcntl;
2704         defcntl_r;
2705         defopen;
2706         defopen_r;
2707         defread;
2708         defread_r;
2709         _delete;
2710         _dgettext;
2711         _doprnt;
2712         _doscan;
2713         _errfp;
2714         _errxfp;
2715         exportfs;
2716         _F_cplx_div;
2717         _F_cplx_div_ix;
2718         _F_cplx_div_rx;
2719         _F_cplx_mul;
2720         __fgetwc_xpg5;
2721         __fgetws_xpg5;
2722         _findbuf;
2723         _findiop;
2724         __fini_daemon_priv;
2725         _finite;
2726         _fork1                  { FLAGS = NODYNSORT };
2727         _forkall                { FLAGS = NODYNSORT };
2728         _fpclass;
2729         _fpgetmask;
2730         _fpgetround;
2731         _fpgetsticky;
2732         _fprintf;
2733         _fpsetmask;
2734         _fpsetround;
2735         _fpsetsticky;
2736         __fputwc_xpg5;
2737         __fputws_xpg5;
2738         _ftw;
2739         _gcvt;
2740         _getarg;
2741         __getcontext;
2742         _getdents;
2743         _get_exit_frame_monitor;
2744         _getfp;
2745         _getgroupsbymember;
2746         _getlogin_r;
2747         _getsp;
2748         __gettsp;
2749         getvmusage;
2750         __getwchar_xpg5;
2751         __getwc_xpg5;
2752         gtty;
2753         __idmap_flush_kcache;
2754         __idmap_reg;
2755         __idmap_unreg;
2756         __init_daemon_priv;
2757         __init_suid_priv;
2758         _insert;
2759         inst_sync;
2760         _iswctype;
2761         klpd_create;
2762         klpd_getpath;
2763         klpd_getport;
2764         klpd_getucred;
2765         klpd_register;
2766         klpd_register_id;
2767         klpd_unregister;
```

```
2768        klpd_unregister_id;
2769        _lgrp_home_fast         { FLAGS = NODYNSORT };
2770        _lgrpsys;
2771        _lltostr;
2772        _lock_clear;
2773        _lock_try;
2774        _ltzset;
2775        lwp_self;
2776        makeut;
2777        makeutx;
2778        _mbftowc;
2779        mcfiller;
2780        mntopt;
2781        modctl;
2782        modutx;
2783        msgctl64;
2784        __multi_innetgr;
2785        _mutex_destroy          { FLAGS = NODYNSORT };
2786        mutex_held;
2787        _mutex_init             { FLAGS = NODYNSORT };
2788        _mutex_unlock           { FLAGS = NODYNSORT };
2789        name_to_attr;
2790        nfs_getfh;
2791        nfssvc;
2792        _nfssys;
2793        __nis_get_environment;
2794        _nss_db_state_destr;
2795        nss_default_key2str;
2796        nss_delete;
2797        nss_endent;
2798        nss_getent;
2799        _nss_initf_group;
2800        _nss_initf_netgroup;
2801        _nss_initf_passwd;
2802        _nss_initf_shadow;
2803        nss_packed_arg_init;
2804        nss_packed_context_init;
2805        nss_packed_getkey;
2806        nss_packed_set_status;
2807        nss_search;
2808        nss_setent;
2809        _nss_XbyY_fgets;
2810        __nsw_extended_action_v1;
2811        __nsw_freeconfig_v1;
2812        __nsw_getconfig_v1;
2813        __nthreads;
2814        __openattrdirat;
2815        option_to_attr;
2816        __priv_bracket;
2817        __priv_relinquish;
2818        pset_assign_forced;
2819        pset_bind_lwp;
2820        _psignal;
2821        _pthread_setcleanupinit;
2822        __putwchar_xpg5;
2823        __putwc_xpg5;
2824        rctlctl;
2825        rctllist;
2826        _realbufend;
2827        _resume;
2828        _resume_ret;
2829        _rpcsys;
2830        _sbrk_grow_aligned;
2831        scrwidth;
2832        semctl64;
2833        _semctl64;
```



```
2834        set_setcontext_enforcement;
2835        _setbufend;
2836        __set_errno;
2837        setprojrctl;
2838        _setregid;
2839        _setreuid;
2840        setsigacthandler;
2841        shmctl64;
2842        _shmctl64;
2843        sigflag;
2844        _signal;
2845        _sigoff;
2846        _sigon;
2847        _so_accept;
2848        _so_bind;
2849        _sockconfig;
2850        _so_connect;
2851        _so_getpeername;
2852        _so_getsockname;
2853        _so_getsockopt;
2854        _so_listen;
2855        _so_recv;
2856        _so_recvfrom;
2857        _so_recvmsg;
2858        _so_send;
2859        _so_sendmsg;
2860        _so_sendto;
2861        _so_setsockopt;
2862        _so_shutdown;
2863        _so_socket;
2864        _so_socketpair;
2865        str2group;
2866        str2passwd;
2867        str2spwd;
2868        __strptime_dontzero;
2869        stty;
2870        syscall;
2871        _sysconfig;
2872        __systemcall;
2873        thr_continue_allmutators;
2874        _thr_continue_allmutators;
2875        thr_continue_mutator;
2876        _thr_continue_mutator;
2877        thr_getstate;
2878        _thr_getstate;
2879        thr_mutators_barrier;
2880        _thr_mutators_barrier;
2881        thr_probe_setup;
2882        _thr_schedctl;
2883        thr_setmutator;
2884        _thr_setmutator;
2885        thr_setstate;
2886        _thr_setstate;
2887        thr_sighndlrinfo;
2888        _thr_sighndlrinfo;
2889        _thr_slot_offset;
2890        thr_suspend_allmutators;
2891        _thr_suspend_allmutators;
2892        thr_suspend_mutator;
2893        _thr_suspend_mutator;
2894        thr_wait_mutator;
2895        _thr_wait_mutator;
2896        __tls_get_addr;
2897        tpool_create;
2898        tpool_dispatch;
2899        tpool_destroy;
```

```
2900         tpool_wait;
2901         tpool_suspend;
2902         tpool_suspended;
2903         tpool_resume;
2904         tpool_member;
2905         _ttyname_dev;
2906         _ucred_alloc;
2907         ucred_getamask;
2908         _ucred_getamask;
2909         ucred_getasid;
2910         _ucred_getasid;
2911         ucred_getatid;
2912         _ucred_getatid;
2913         ucred_getauid;
2914         _ucred_getauid;
2915         _ulltostr;
2916         _uncached_getgrgid_r;
2917         _uncached_getgrnam_r;
2918         _uncached_getpwnam_r;
2919         _uncached_getpwuid_r;
2920         __ungetwc_xpg5;
2921         _unordered;
2922         utssys;
2923         _verrfp;
2924         _verrxfp;
2925         _vwarnfp;
2926         _vwarnxfp;
2927         _warnfp;
2928         _warnxfp;
2929         __wcsftime_xpg5;
2930         __wcstok_xpg5;
2931         wdbindf;
2932         wdchkind;
2933         wddelim;
2934         _wrtchk;
2935         _xflsbuf;
2936         _xgetwidth;
2937         zone_add_datalink;
2938         zone_boot;
2939         zone_check_datalink;
2940         zone_create;
2941         zone_destroy;
2942         zone_enter;
2943         zone_getattr;
2944         zone_get_id;
2945         zone_list;
2946         zone_list_datalink;
2947         zonept;
2948         zone_remove_datalink;
2949         zone_setattr;
2950         zone_shutdown;
2951         zone_version;

2953 $if _ELF32
2954         __divdi3;
2955         _file_set;
2956         _fprintf_c89;
2957         _fscanf_c89;
2958         _fwprintf_c89;
2959         _fwscanf_c89;
2960         _imaxabs_c89;
2961         _imaxdiv_c89;
2962         __moddi3;
2963         _printf_c89;
2964         _scanf_c89;
2965         _snprintf_c89;
```



```
2966         _sprintf_c89;
2967         _sscanf_c89;
2968         _strtoimax_c89;
2969         _strtoumax_c89;
2970         _swprintf_c89;
2971         _swscanf_c89;
2972         __udivdi3;
2973         __umoddi3;
2974         _vfprintf_c89;
2975         _vfscanf_c89;
2976         _vfwprintf_c89;
2977         _vfwscanf_c89;
2978         _vprintf_c89;
2979         _vscanf_c89;
2980         _vsnprintf_c89;
2981         _vsprintf_c89;
2982         _vsscanf_c89;
2983         _vswprintf_c89;
2984         _vswscanf_c89;
2985         _vwprintf_c89;
2986         _vwscanf_c89;
2987         _wcstoimax_c89;
2988         _wcstoumax_c89;
2989         _wprintf_c89;
2990         _wscanf_c89;
2991 $endif

2993 $if _sparc
2994         _cerror;
2995         install_utrap;
2996         _install_utrap;
2997         nop;
2998         _Q_cplx_div;
2999         _Q_cplx_div_ix;
3000         _Q_cplx_div_rx;
3001         _Q_cplx_lr_div;
3002         _Q_cplx_lr_div_ix;
3003         _Q_cplx_lr_div_rx;
3004         _Q_cplx_lr_mul;
3005         _Q_cplx_mul;
3006         _QgetRD;
3007         _xregs_clrptr;
3008 $endif

3010 $if sparc32
3011         __ashldi3;
3012         __ashrdi3;
3013         _cerror64;
3014         __cmpdi2;
3015         __floatdidf;
3016         __floatdisf;
3017         __floatundidf;
3018         __floatundisf;
3019         __lshrdi3;
3020         __muldi3;
3021         __ucmpdi2;
3022 $endif

3024 $if _x86
3025         _D_cplx_lr_div;
3026         _D_cplx_lr_div_ix;
3027         _D_cplx_lr_div_rx;
3028         _F_cplx_lr_div;
3029         _F_cplx_lr_div_ix;
3030         _F_cplx_lr_div_rx;
3031         __fltrounds;
```

```
3032         sysi86;
3033         _sysi86;
3034         _X_cplx_div;
3035         _X_cplx_div_ix;
3036         _X_cplx_div_rx;
3037         _X_cplx_lr_div;
3038         _X_cplx_lr_div_ix;
3039         _X_cplx_lr_div_rx;
3040         _X_cplx_mul;
3041         __xgetRD;
3042         __xtol;
3043         __xtoll;
3044         __xtoul;
3045         __xtoull;
3046 $endif

3048 $if i386
3049         __divrem64;
3050         ___tls_get_addr;
3051         __udivrem64;
3052 $endif

3054 # The following functions should not be exported from libc,
3055 # but /lib/libm.so.2, some older versions of the Studio
3056 # compiler/debugger components, and some ancient programs
3057 # found in /usr/dist reference them.  When we no longer
3058 # care about these old and broken binary objects, these
3059 # symbols should be deleted.
3060         _brk                                { FLAGS = NODYNSORT };
3061         _cond_broadcast                     { FLAGS = NODYNSORT };
3062         _cond_init                          { FLAGS = NODYNSORT };
3063         _cond_signal                        { FLAGS = NODYNSORT };
3064         _cond_wait                          { FLAGS = NODYNSORT };
3065         _ecvt                               { FLAGS = NODYNSORT };
3066         _fcvt                               { FLAGS = NODYNSORT };
3067         _getc_unlocked                      { FLAGS = NODYNSORT };
3068         _llseek                             { FLAGS = NODYNSORT };
3069         _pthread_attr_getdetachstate        { FLAGS = NODYNSORT };
3070         _pthread_attr_getinheritsched       { FLAGS = NODYNSORT };
3071         _pthread_attr_getschedparam         { FLAGS = NODYNSORT };
3072         _pthread_attr_getschedpolicy        { FLAGS = NODYNSORT };
3073         _pthread_attr_getscope              { FLAGS = NODYNSORT };
3074         _pthread_attr_getstackaddr          { FLAGS = NODYNSORT };
3075         _pthread_attr_getstacksize          { FLAGS = NODYNSORT };
3076         _pthread_attr_init                  { FLAGS = NODYNSORT };
3077         _pthread_condattr_getpshared        { FLAGS = NODYNSORT };
3078         _pthread_condattr_init              { FLAGS = NODYNSORT };
3079         _pthread_cond_init                  { FLAGS = NODYNSORT };
3080         _pthread_create                     { FLAGS = NODYNSORT };
3081         _pthread_getschedparam              { FLAGS = NODYNSORT };
3082         _pthread_join                       { FLAGS = NODYNSORT };
3083         _pthread_key_create                 { FLAGS = NODYNSORT };
3084         _pthread_mutexattr_getprioceiling   { FLAGS = NODYNSORT };
3085         _pthread_mutexattr_getprotocol      { FLAGS = NODYNSORT };
3086         _pthread_mutexattr_getpshared       { FLAGS = NODYNSORT };
3087         _pthread_mutexattr_init             { FLAGS = NODYNSORT };
3088         _pthread_mutex_getprioceiling       { FLAGS = NODYNSORT };
3089         _pthread_mutex_init                 { FLAGS = NODYNSORT };
3090         _pthread_sigmask                    { FLAGS = NODYNSORT };
3091         _rwlock_init                        { FLAGS = NODYNSORT };
3092         _rw_rdlock                          { FLAGS = NODYNSORT };
3093         _rw_unlock                          { FLAGS = NODYNSORT };
3094         _rw_wrlock                          { FLAGS = NODYNSORT };
3095         _sbrk_unlocked                      { FLAGS = NODYNSORT };
3096         _select                             { FLAGS = NODYNSORT };
3097         _sema_init                          { FLAGS = NODYNSORT };
```



```
3098         _sema_post                          { FLAGS = NODYNSORT };
3099         _sema_trywait                       { FLAGS = NODYNSORT };
3100         _sema_wait                          { FLAGS = NODYNSORT };
3101         _sysfs                              { FLAGS = NODYNSORT };
3102         _thr_create                         { FLAGS = NODYNSORT };
3103         _thr_exit                           { FLAGS = NODYNSORT };
3104         _thr_getprio                        { FLAGS = NODYNSORT };
3105         _thr_getspecific                    { FLAGS = NODYNSORT };
3106         _thr_join                           { FLAGS = NODYNSORT };
3107         _thr_keycreate                      { FLAGS = NODYNSORT };
3108         _thr_kill                           { FLAGS = NODYNSORT };
3109         _thr_main                           { FLAGS = NODYNSORT };
3110         _thr_self                           { FLAGS = NODYNSORT };
3111         _thr_setspecific                    { FLAGS = NODYNSORT };
3112         _thr_sigsetmask                     { FLAGS = NODYNSORT };
3113         _thr_stksegment                     { FLAGS = NODYNSORT };
3114         _ungetc_unlocked                    { FLAGS = NODYNSORT };

3116     local:
3117         __imax_lldiv                        { FLAGS = NODYNSORT };
3118         _ti_thr_self                        { FLAGS = NODYNSORT };
3119         *;

3121 $if lf64
3122         _seekdir64              { FLAGS = NODYNSORT };
3123         _telldir64              { FLAGS = NODYNSORT };
3124 $endif

3126 $if _sparc
3127         __cerror                { FLAGS = NODYNSORT };
3128 $endif

3130 $if sparc32
3131         __cerror64              { FLAGS = NODYNSORT };
3132 $endif

3134 $if sparcv9
3135         __cleanup               { FLAGS = NODYNSORT };
3136 $endif

3138 $if i386
3139         _syscall6               { FLAGS = NODYNSORT };
3140         __systemcall6           { FLAGS = NODYNSORT };
3141 $endif

3143 $if amd64
3144         ___tls_get_addr         { FLAGS = NODYNSORT };
3145 $endif
3146 };
```

```
**********************************************************
     519 Thu Jul 11 12:26:15 2013
new/usr/src/test/os-tests/tests/Makefile
2964 need POSIX 2008 locale object support
**********************************************************
   1 #
   2 # This file and its contents are supplied under the terms of the
   3 # Common Development and Distribution License ("CDDL"), version 1.0.
   4 # You may only use this file in accordance with the terms of version
   5 # 1.0 of the CDDL.
   6 #
   7 # A full copy of the text of the CDDL should have accompanied this
   8 # source.  A copy of the CDDL is also available via the Internet at
   9 # http://www.illumos.org/license/CDDL.
  10 #

  12 #
  13 # Copyright (c) 2012 by Delphix. All rights reserved.
  14 #

  16 SUBDIRS = poll xlocale
  16 SUBDIRS = poll

  18 include $(SRC)/test/Makefile.com
```

```
**********************************************************
    1050 Thu Jul 11 12:26:16 2013
new/usr/src/test/os-tests/tests/xlocale/Makefile
2964 need POSIX 2008 locale object support
**********************************************************
```

```
   1 #
   2 # This file and its contents are supplied under the terms of the
   3 # Common Development and Distribution License ("CDDL"), version 1.0.
   4 # You may only use this file in accordance with the terms of version
   5 # 1.0 of the CDDL.
   6 #
   7 # A full copy of the text of the CDDL should have accompanied this
   8 # source.  A copy of the CDDL is also available via the Internet at
   9 # http://www.illumos.org/license/CDDL.
  10 #

  12 #
  13 # Copyright (c) 2012 by Delphix. All rights reserved.
  14 #

  16 include $(SRC)/cmd/Makefile.cmd
  17 include $(SRC)/test/Makefile.com

  19 PROG = xlocale_test
  20 OBJS = $(PROG:%=%.o)
  21 SRCS = $(OBJS:%.o=%.c)

  23 #LDLIBS += -lsocket
  24 C99MODE = -xc99=%all

  26 ROOTOPTPKG = $(ROOT)/opt/os-tests
  27 TESTDIR = $(ROOTOPTPKG)/tests

  29 CMDS = $(PROG:%=$(TESTDIR)/%)
  30 $(CMDS) := FILEMODE = 0555

  32 all: $(PROG)

  34 $(PROG): $(OBJS)
  35         $(LINK.c) $(OBJS) -o $@ $(LDLIBS)
  36         $(POST_PROCESS)

  38 %.o: ../%.c
  39         $(COMPILE.c) $<

  41 install: all $(CMDS)

  43 lint: lint_SRCS

  45 clobber: clean
  46         -$(RM) $(PROG)

  48 clean:
  49         -$(RM) $(OBJS)

  51 $(CMDS): $(TESTDIR) $(PROG)

  53 $(TESTDIR):
  54         $(INS.dir)

  56 $(TESTDIR)/%: %
  57         $(INS.file)
  58 #endif /* ! codereview */
```

```
**********************************************************
     809 Thu Jul 11 12:26:16 2013
new/usr/src/test/os-tests/tests/xlocale/xlocale_test.c
2964 need POSIX 2008 locale object support
**********************************************************
```

```
   1 /*
   2  * This file and its contents are supplied under the terms of the
   3  * Common Development and Distribution License ("CDDL"), version 1.0.
   4  * You may only use this file in accordance with the terms of version
   5  * 1.0 of the CDDL.
   6  *
   7  * A full copy of the text of the CDDL should have accompanied this
   8  * source.  A copy of the CDDL is also available via the Internet at
   9  * http://www.illumos.org/license/CDDL.
  10  */

  12 /*
  13  * Copyright 2013 David Hoeppner.  All rights reserved.
  14  */

  16 #include <locale.h>
  17 #include <stdlib.h>

  19 static void
  20 newlocale_test(void)
  21 {
  22         locale_t        new_loc = NULL;

  24         new_loc = newlocale(LC_CTYPE_MASK, "loc1", (locale_t)NULL);
  25         if (new_loc == (locale_t)NULL)
  26                 return;
  27 }

  29 static void
  30 run_tests(void)
  31 {
  32         newlocale_test();
  33 }

  35 int
  36 main(int argc, char * const argv[])
  37 {

  39         run_tests();

  41         exit(0);
  42 }
  43 #endif /* ! codereview */
```