

```

*****
11469 Tue Sep 3 18:09:39 2013
new/usr/src/cmd/Makefile
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 #
22 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
23 # Copyright 2010 Nexenta Systems, Inc. All rights reserved.
24 # Copyright 2011 Joyent, Inc. All rights reserved.
25 # Copyright (c) 2012 by Delphix. All rights reserved.
26 # Copyright (c) 2013 DEY Storage Systems, Inc. All rights reserved.
27 #
28 include ../Makefile.master
29 #
30 #
31 # Note that the commands 'agents', 'lp', 'perl', and 'man' are first in
32 # the list, violating alphabetical order. This is because they are very
33 # long-running and should be given the most wall-clock time for a
34 # parallel build.
35 #
36 # Commands in the FIRST_SUBDIRS list are built before starting the build
37 # of other commands. Currently this includes only 'isaexec' and
38 # 'platexec'. This is necessary because $(ROOT)/usr/lib/isaexec or
39 # $(ROOT)/usr/lib/platexec must exist when some other commands are built
40 # because their 'make install' creates a hard link to one of them.
41 #
42 # Commands are listed one per line so that TeamWare can auto-merge most
43 # changes.
44 #
45 #
46 FIRST_SUBDIRS= \
47 isaexec \
48 platexec
49 #
50 COMMON_SUBDIRS= \
51 allocate \
52 availdevs \
53 lp \
54 perl \
55 man \
56 Adm \
57 abi \
58 adbgen \
59 acct \
60 acctadm \
61 arch

```

```

62 asa \
63 ast \
64 audio \
65 auths \
66 autopush \
67 avs \
68 awk \
69 awk_xpg4 \
70 backup \
71 banner \
72 bart \
73 basename \
74 bc \
75 bdiff \
76 beadm \
77 bfs \
78 bnu \
79 boot \
80 busstat \
81 cal \
82 calendar \
83 captinfo \
84 cat \
85 cdrw \
86 cfgadm \
87 checkeq \
88 checknr \
89 chgrp \
90 chmod \
91 chown \
92 chroot \
93 clear \
94 clinfo \
95 cmd-crypto \
96 cmd-inet \
97 col \
98 compress \
99 consadm \
100 coreadm \
101 cpio \
102 cpc \
103 cron \
104 crypt \
105 csh \
106 csplit \
107 ctrun \
108 ctstat \
109 ctwatch \
110 datadm \
111 date \
112 dc \
113 dd \
114 deroff \
115 devfsadm \
116 syseventd \
117 devctl \
118 devinfo \
119 devmgmt \
120 devprop \
121 dfs.cmds \
122 diff \
123 diff3 \
124 diffmk \
125 dircmp \
126 dirname \
127 dis \

```

## new/usr/src/cmd/Makefile

```

128      diskmgtd //
129      dispadmin //
130      dladm //
131      dlstat //
132      dmesg //
133      dodatadm //
134      dtrace //
135      du //
136      dumpadm //
137      dumpcs //
138      echo //
139      ed //
140      eeprom //
141      egrep //
142      eject //
143      emul64ioctl //
144      enhance //
145      env //
146      eqn //
147      expand //
148      expr //
149      exstr //
150      factor //
151      false //
152      fcinfo //
153      fcoesvc //
154      fdetach //
155      fdformat //
156      fdisk //
157      filesync //
158      fgrep //
159      file //
160      filebench //
161      find //
162      flowadm //
163      flowstat //
164      fm //
165      fmt //
166      fmthard //
167      fmtnsg //
168      fold //
169      format //
170      fs.d //
171      fsdadm //
172      fstyp //
173      fuser //
174      fwflash //
175      gcore //
176      gencat //
177      geniconvtbl //
178      genmsg //
179      getconf //
180      getdevpolicy //
181      getent //
182      getfacl //
183      getmajor //
184      getopt //
185      gettext //
186      gettxt //
187      grep //
188      grep_xpg4 //
189      groups //
190      grpck //
191      gss //
192      hal //
193      halt //

```

3

## new/usr/src/cmd/Makefile

```

194      head //
195      hostid //
196      hostname //
197      hotplug //
198      hotplugd //
199      hwdata //
200      ibd_upgrade //
201      id //
202      idmap //
203      infocmp //
204      init //
205      initpkg //
206      install.d //
207      intrd //
208      intrstat //
209      ipcrm //
210      ipcs //
211      ipf //
212      isainfo //
213      isalist //
214      itutools //
215      iscsiadm //
216      iscsid //
217      iscsitsvc //
218      isns //
219      itadm //
220      java //
221      kbd //
222      keyserv //
223      killall //
224      krb5 //
225      ksh //
226      kvmstat //
227      last //
228      lastcomm //
229      latencytop //
230      ldap //
231      ldapcachemgr //
232      lgrpinfo //
233      line //
234      link //
235      dlmgmt //
236      listen //
237      loadkeys //
238      locale //
239      localedef //
240      lockstat //
241      locator //
242      lofiadm //
243      logadm //
244      logger //
245      login //
246      logins //
247      look //
248      ls //
249      luxadm //
250      lvm //
251      mach //
252      machid //
253      mail //
254      mailx //
255      makekey //
256      mdb //
257      msg //
258      mkdir //
259      mkfifo //

```

4

new/usr/src/cmd/Makefile

260 mkfile //  
261 mkmsgs //  
262 mknod //  
263 mkpwdict //  
264 mktemp //  
265 modload //  
266 more //  
267 mpathadm //  
268 msgfmt //  
269 msgid //  
270 mt //  
271 mv //  
272 mvdir //  
273 ndmpadm //  
274 ndmpd //  
275 ndmpstat //  
276 netadm //  
277 netfiles //  
278 newform //  
279 newgrp //  
280 news //  
281 newtask //  
282 nice //  
283 nl //  
284 nlsadmin //  
285 nohup //  
286 nsadmin //  
287 nscd //  
288 oamuser //  
289 oawk //  
290 od //  
291 pack //  
292 pagesize //  
293 passmgmt //  
294 passwd //  
295 pathchk //  
296 pbind //  
297 pcidr //  
298 pcitool //  
299 pfexec //  
300 pfexecd //  
301 pginfo //  
302 pgstat //  
303 pgrep //  
304 picl //  
305 plimit //  
306 policykit //  
307 pools //  
308 power //  
309 powertop //  
310 ppgsz //  
311 pg //  
312 plockstat //  
313 pr //  
314 prctl //  
315 print //  
316 printf //  
317 priocntl //  
318 profiles //  
319 projadd //  
320 projects //  
321 prstat //  
322 prtconf //  
323 prtdiag //  
324 prtvtoc //  
325 ps //

new/usr/src/cmd/Makefile

326 psradm //  
327 psrinfo //  
328 psrset //  
329 ptools //  
330 pwck //  
331 pwconv //  
332 pwd //  
333 pyzfs //  
334 raidctl //  
335 ramdiskadm //  
336 rcap //  
337 rcm\_daemon //  
338 rctladm //  
339 refer //  
340 regcmp //  
341 renice //  
342 rexd //  
343 rm //  
344 rmdir //  
345 rmformat //  
346 rmmount //  
347 rmt //  
348 rmvolmgr //  
349 roles //  
350 rpcbind //  
351 rpcgen //  
352 rpcinfo //  
353 rpcsvc //  
354 runat //  
355 sa //  
356 saf //  
357 sasinfo //  
358 savecore //  
359 sbdadm //  
360 script //  
361 scsi //  
362 sdiff //  
363 sdpadm //  
364 sed //  
365 sendmail //  
366 setfacl //  
367 setmnt //  
368 setpgrp //  
369 setuname //  
370 sgs //  
371 sh //  
372 shcomp //  
373 smbios //  
374 smbstrv //  
375 smserverd //  
376 soelim //  
377 sort //  
378 spell //  
379 split //  
380 sqlite //  
381 srchtxt //  
382 srptadm //  
383 srptsvc //  
384 ssh //  
385 stat //  
386 stmfadm //  
387 stmfproxy //  
388 stmfsvc //  
389 stmsboot //  
390 streams //  
391 strings //

## new/usr/src/cmd/Makefile

```

392      su                \
393      sulogin           \
394      sunpc             \
395      svc               \
396      svr4pkg          \
397      swap              \
398      sync              \
399      sysdef            \
400      syseventadm      \
401      syslogd          \
402      tabs             \
403      tail              \
404      tar               \
405      tbl               \
406      tcopy            \
407      tcpd             \
408      terminfo         \
409      th_tools         \
410      tic               \
411      time              \
412      tip               \
413      tnf               \
414      touch            \
415      tput              \
416      tr                \
417      trapstat        \
418      troff            \
419      true              \
420      truss            \
421      tsol              \
422      tty               \
423      ttymon           \
424      tzreload         \
425      uadmin           \
426      ul                \
427      uname            \
428      units            \
429      unlink           \
430      unpack           \
431      userattr         \
432      users            \
433      utmp_update      \
434      utmpd            \
435      valtools         \
436      vgrind           \
437      vi                \
438      volcheck         \
439      volrmount        \
440      vrrpadm          \
441      vscan            \
442      vt                \
443      w                 \
444      wall              \
445      which             \
446      who               \
447      whodo            \
448      wracct           \
449      write            \
450      wusbadm          \
451      xargs            \
452      xstr              \
453      yes               \
454      ypcmd            \
455      yppasswd         \
456      zdb               \
457      zdump            \

```

7

## new/usr/src/cmd/Makefile

```

458      zfs               \
459      zhack             \
460      zic               \
461      zinject          \
462      zlogin           \
463      zoneadm          \
464      zoneadmd         \
465      zonecfg          \
466      zonename         \
467      zpool            \
468      zlook            \
469      zonestat         \
470      zstreamdump     \
471      ztest            \

473 $(CLOSED_BUILD)COMMON_SUBDIRS += \
474     $(CLOSED)/cmd/iconv \
475     $(CLOSED)/cmd/ksh   \
476     $(CLOSED)/cmd/localedef \
477     $(CLOSED)/cmd/more_xpg4 \
478     $(CLOSED)/cmd/mtst \
479     $(CLOSED)/cmd/od    \
480     $(CLOSED)/cmd/patch \
481     $(CLOSED)/cmd/pax   \
482     $(CLOSED)/cmd/printf \
483     $(CLOSED)/cmd/sed   \
484     $(CLOSED)/cmd/sed_xpg4 \

486 i386_SUBDIRS= \
487     acpihpd \
488     addbadsec \
489     biosdev \
490     diskscan \
491     lms \
492     ntfsprops \
493     parted \
494     rtc \
495     ucodeadm \
496     xvm \

498 sparc_SUBDIRS= \
499     cvcd \
500     dcs \
501     device_remap \
502     drd \
503     fruadm \
504     ldmad \
505     oplhpd \
506     prtdscp \
507     prtfru \
508     scadm \
509     sckmd \
510     sf880drd \
511     virtinfo \
512     vntsd \

514 #
515 # Commands that are messaged. Note that 'lp' and 'man' come first
516 # (see previous comment about 'lp' and 'man').
517 #
518 MSGSUBDIRS= \
519     lp \
520     man \
521     abi \
522     acctadm \
523     allocate \

```

8

new/usr/src/cmd/Makefile

```
524      asa //
525      audio //
526      audit //
527      auditconfig //
528      auditd //
529      auditrecord //
530      auditset //
531      auths //
532      autopush //
533      avs //
534      awk //
535      awk_xpg4 //
536      backup //
537      banner //
538      bart //
539      basename //
540      beadm //
541      bnu //
542      busstat //
543      cal //
544      cat //
545      cdrw //
546      cfgadm //
547      checkeq //
548      checknr //
549      chgrp //
550      chmod //
551      chown //
552      cmd-crypto //
553      cmd-inet //
554      col //
555      compress //
556      consadm //
557      coreadm //
558      cpio //
559      cpc //
560      cron //
561      csh //
562      csplit //
563      ctrun //
564      ctstat //
565      ctwatch //
566      datadm //
567      date //
568      dc //
569      dcs //
570      dd //
571      deroff //
572      devfsadm //
573      dfs.cmds //
574      diff //
575      diffmk //
576      dladm //
577      dlstat //
578      du //
579      dumpcs //
580      ed //
581      eject //
582      env //
583      eqn //
584      expand //
585      expr //
586      fcinfo //
587      fgrep //
588      file //
589      filesync //
```

new/usr/src/cmd/Makefile

```
590      find //
591      flowadm //
592      flowstat //
593      fm //
594      fold //
595      fs.d //
596      fwflash //
597      geniconvtbl //
598      genmsg //
599      getconf //
600      getent //
601      gettext //
602      gettxt //
603      grep //
604      grep_xpg4 //
605      grpck //
606      gss //
607      halt //
608      head //
609      hostname //
610      hotplug //
611      id //
612      idmap //
613      isaexec //
614      iscsiadm //
615      iscsid //
616      isns //
617      itadm //
618      kbd //
619      krb5 //
620      ksh //
621      last //
622      ldap //
623      ldapcachemgr //
624      lgrpinfo //
625      locale //
626      lofiadm //
627      logadm //
628      logger //
629      logins //
630      ls //
631      luxadm //
632      lvm //
633      mailx //
634      mesg //
635      mkdir //
636      mkpwdict //
637      mktemp //
638      more //
639      mpathadm //
640      msgfmt //
641      mv //
642      ndmpadm //
643      ndmpstat //
644      newgrp //
645      newtask //
646      nice //
647      nohup //
648      oawk //
649      pack //
650      passwd //
651      passmgmt //
652      pathchk //
653      pfexec //
654      pg //
655      pgrep //
```

```

656      picl          \
657      pools        \
658      power        \
659      pr           \
660      praudit      \
661      print        \
662      profiles     \
663      projadd      \
664      projects     \
665      prstat       \
666      prtdiag     \
667      ps           \
668      psrinfo      \
669      ptools       \
670      pwconv       \
671      pwd          \
672      pyzfs        \
673      raidctl     \
674      ramdiskadm  \
675      rcap         \
676      rcm_daemon  \
677      refer        \
678      regcmp       \
679      renice       \
680      roles        \
681      rm           \
682      rmdir        \
683      rmformat    \
684      rmmount     \
685      rmvolmgr    \
686      sasinfo      \
687      sbdadm       \
688      scadm        \
689      script       \
690      scsi         \
691      sdiff        \
692      sdpadm       \
693      sgs          \
694      sh           \
695      shcomp       \
696      smbsrv       \
697      sort         \
698      split        \
699      srptadm     \
700      ssh          \
701      stat         \
702      stmfadm     \
703      stmsboot    \
704      strings      \
705      su           \
706      svc          \
707      svr4pkg     \
708      swap         \
709      syseventadm \
710      syseventd   \
711      tabs        \
712      tar          \
713      tbl         \
714      time        \
715      tnf         \
716      touch       \
717      tput        \
718      troff       \
719      tsol        \
720      tty         \
721      ttymon      \

```

```

722      tzreload    \
723      ul          \
724      uname       \
725      units       \
726      unlink      \
727      unpack      \
728      userattr    \
729      valtools    \
730      vgrind      \
731      vi          \
732      volcheck    \
733      volrmmount \
734      vrrpadm     \
735      vscan       \
736      w           \
737      who         \
738      whodo       \
739      wracct      \
740      write       \
741      wusbadm     \
742      xargs       \
743      yppasswd    \
744      zdump       \
745      zfs         \
746      zic         \
747      zlogin      \
748      zoneadm     \
749      zoneadmd    \
750      zonecfg     \
751      zonename    \
752      zpool       \
753      zonestat    \
754      \
755      $(CLOSED_BUILD)MSGSUBDIRS += \
756      $(CLOSED)/cmd/iconv          \
757      $(CLOSED)/cmd/ksh            \
758      $(CLOSED)/cmd/localedef     \
759      $(CLOSED)/cmd/more_xpg4     \
760      $(CLOSED)/cmd/od             \
761      $(CLOSED)/cmd/patch         \
762      $(CLOSED)/cmd/pax            \
763      $(CLOSED)/cmd/printf        \
764      $(CLOSED)/cmd/sed           \
765      $(CLOSED)/cmd/sed_xpg4     \
766      \
767      sparc_MSGSUBDIRS=           \
768      fruadm                      \
769      prtdscp                     \
770      prtfru                       \
771      virtinfo                     \
772      vntsd                        \
773      \
774      i386_MSGSUBDIRS=           \
775      ucodeadm                    \
776      \
777      #
778      # commands that use dcgettext for localized time, LC_TIME
779      #
780      DCSUBDIRS=                  \
781      cal                         \
782      cfgadm                      \
783      diff                         \
784      ls                           \
785      pr                           \
786      ps                           \
787      tar                          \

```

## new/usr/src/cmd/Makefile

```

788      w          \
789      who        \
790      whodo     \
791      write

793 $(CLOSED_BUILD)DCSUBDIRS += \
794   $(CLOSED)/cmd/pax

796 #
797 # commands that belong only to audit.
798 #
799 AUDITSUBDIRS=      \
800     amt             \
801     audit          \
802     audit_warn     \
803     auditconfig    \
804     auditd         \
805     auditrecord    \
806     auditreduce    \
807     auditset       \
808     auditstat      \
809     praudit

811 #
812 # commands not owned by the systems group
813 #
814 BWOSDIRS=

817 all :=          TARGET = all
818 install :=      TARGET = install
819 clean :=        TARGET = clean
820 clobber :=     TARGET = clobber
821 lint :=         TARGET = lint
822 _msg :=         TARGET = _msg
823 _dc :=          TARGET = _dc

825 .KEEP_STATE:

827 SUBDIRS = $(COMMON_SUBDIRS) $( $(MACH)_SUBDIRS )

829 .PARALLEL:      $(BWOSDIRS) $(SUBDIRS) $(MSGSUBDIRS) $(AUDITSUBDIRS)

831 all install clean clobber lint: $(FIRST_SUBDIRS) .WAIT $(SUBDIRS) \
832   $(AUDITSUBDIRS)

834 #
835 # Manifests cannot be checked in parallel, because we are using
836 # the global repository that is in $(SRC)/cmd/svc/seed/global.db.
837 # For this reason, to avoid .PARALLEL and .NO_PARALLEL conflicts,
838 # we spawn off a sub-make to perform the non-parallel 'make check'
839 #
840 check:
841     $(MAKE) -f Makefile.check check

843 #
844 # The .WAIT directive works around an apparent bug in parallel make.
845 # Evidently make was getting the target _msg vs. _dc confused under
846 # some level of parallelization, causing some of the _dc objects
847 # not to be built.
848 #
849 _msg: $(MSGSUBDIRS) $( $(MACH)_MSGSUBDIRS ) .WAIT _dc

851 _dc: $(DCSUBDIRS)

853 #

```

13

## new/usr/src/cmd/Makefile

```

854 # Dependencies
855 #
856 fs.d: fstyp
857 ksh:   shcomp isaexec
858 mdb:   terminfo
859 print: lp

861 $(FIRST_SUBDIRS) $(BWOSDIRS) $(SUBDIRS) $(AUDITSUBDIRS): FRC
862     @if [ -f $@/Makefile ]; then \
863         cd $@; pwd; $(MAKE) $(TARGET); \
864     else \
865         true; \
866     fi

868 FRC:

```

14

new/usr/src/cmd/fsdadm/Makefile

1

\*\*\*\*\*

904 Tue Sep 3 18:09:39 2013

new/usr/src/cmd/fsdadm/Makefile

fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 #
11 #
12 #
13 # Copyright 2013 Damian Bogel. All rights reserved.
14 #
15 #
16 PROG = fsdadm
17 OBJS = fsdadm.o
18 SRCS = $(OBJS:%.o=%.c)
19 #
20 FILEMODE = 0555
21 #
22 include ../Makefile.cmd
23 include ../Makefile.ctf
24 #
25 CLEANFILES += $(OBJS)
26 #
27 CFLAGS += $(CCVERBOSE)
28 CFLAG64 += $(CCVERBOSE)
29 #
30 LDLIBS += -lfsd
31 #
32 all: $(PROG)
33 #
34 $(PROG): $(OBJS)
35     $(LINK.c) -o $@ $(OBJS) $(LDLIBS)
36     $(POST_PROCESS)
37 #
38 %.o: %.c
39     $(COMPILE.c) $<
40     $(POST_PROCESS_O)
41 #
42 clean:
43     -$(RM) $(CLEANFILES)
44 #
45 lint: lint_PROG
46 #
47 install: all $(ROOTUSRSBINPROG)
48 #
49 include ../Makefile.targ
```



```

*****
5450 Tue Sep 3 18:09:39 2013
new/usr/src/cmd/fsdadm/fsdadm.c
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel. All rights reserved.
14  */

16 #include <libfsd.h>
17 #include <stdio.h>
18 #include <stdlib.h>
19 #include <string.h>

21 fsd_handle_t handle;

23 int ret;

25 int
26 errout(fsd_handle_t *handle)
27 {
28     (void) fprintf(stderr, "Error: %s: %s\n",
29                 fsd_strerror(handle->fsd_errno), strerror(handle->errno));

31     return (-1);
32 }

34 void
35 print_fsd(fsd_t *fsd)
36 {
37     (void) printf("\tRead less: %d%% chance with range %d - %d\n",
38                 (int)fsd->read_less_chance,
39                 (int)fsd->read_less_r[0],
40                 (int)fsd->read_less_r[1]);
41 }

43 void
44 info()
45 {
46     fsd_info_t info;

48     if (fsd_get_info(&handle, &info) != 0) {
49         ret = errout(&handle);
50         return;
51     }

53     if (info.fsdinf_enabled) {
54         (void) printf(
55             "Enabled: yes\n"
56             "Filesystems disturbed: %d\n", (int)info.fsdinf_count);

58         if (info.fsdinf_omni_on) {
59             (void) printf(
60                 "Omnipresent disturbing: yes\n"
61                 "Omnipresent params:\n");

```

```

62         print_fsd(&info.fsdinf_omni_param);
63     } else {
64         (void) printf("Omnipresent disturbing: no\n");
65     }
66 }
67 }

69 void
70 list()
71 {
72     fsd_info_t info;
73     fsd_fs_t *fslistp;
74     int i;
75     int count;

77     if (fsd_get_info(&handle, &info) != 0) {
78         ret = errout(&handle);
79         return;
80     }
81     count = info.fsdinf_count;

83     fslistp = calloc(info.fsdinf_count, sizeof (fsd_fs_t));
84     if (fsd_get_list(&handle, fslistp, &count) != 0) {
85         ret = errout(&handle);
86     } else {
87         for (i = 0; i < count; i++) {
88             (void) printf("Mountpoint: %s\n",
89                         fslistp[i].fsdf_name);
90             print_fsd(&fslistp[i].fsdf_param);
91             (void) printf("\n");
92         }
93     }
94 }

96     free(fslistp);
97 }

99 int aflag;
100 int cflag;
101 int dflag;
102 int eflag;
103 int gflag;
104 int iflag;
105 int lflag;
106 int mflag;
107 int oflag;
108 int rflag;
109 int xflag;

111 char *mnt;
112 fsd_t param;
113 int chance;
114 int range[2];

116 int
117 main(int argc, char *argv[])
118 {
119     extern char *optarg;
120     extern int optind;
121     int opt;

123     if (argc < 2)
124         goto usage;

126     if (fsd_open(&handle) != 0)
127         return (errout(&handle));

```

```

129     while ((opt = getopt(argc, argv, "ediam:gxoc:r:l")) != -1) {
130         switch (opt) {
131             case 'e':
132                 eflag = 1;
133                 break;
134
135             case 'd':
136                 dflag = 1;
137                 break;
138
139             case 'i':
140                 iflag = 1;
141                 break;
142
143             case 'a':
144                 aflag = 1;
145                 break;
146
147             case 'm':
148                 mflag = 1;
149                 mnt = optarg;
150                 break;
151
152             case 'g':
153                 gflag = 1;
154                 break;
155
156             case 'x':
157                 xflag = 1;
158                 break;
159
160             case 'o':
161                 oflag = 1;
162                 break;
163
164             case 'c':
165                 cflag = 1;
166                 chance = atoi(optarg);
167                 break;
168
169             case 'r':
170                 rflag = 1;
171                 if (optind > argc - 1) {
172                     (void) fprintf(stderr,
173                         "Error: -r requires two arguments\n");
174                     ret = -1;
175                     goto end;
176                 }
177                 range[0] = atoi(argv[optind-1]);
178                 range[1] = atoi(argv[optind]);
179                 optind++;
180                 break;
181
182             case 'l':
183                 lflag = 1;
184                 break;
185
186             case '?':
187                 (void) fprintf(stderr,
188                     "Error: Unrecognized option: -%c\n", optopt);
189                 ret = -1;
190                 goto end;
191             }
192         }

```

```

194         if (eflag) {
195             if (fsd_enable(&handle) != 0)
196                 ret = errout(&handle);
197
198         } else if (dflag) {
199             if (fsd_disable(&handle) != 0)
200                 ret = errout(&handle);
201
202         } else if (iflag) {
203             info();
204
205         } else if (aflag) {
206             list();
207
208         } else if (xflag) {
209             if (oflag) {
210                 if (fsd_disturb_omni_off(&handle) != 0)
211                     ret = errout(&handle);
212
213             } else if (mflag) {
214                 if (fsd_disturb_off(&handle, mnt) != 0)
215                     ret = errout(&handle);
216
217             } else {
218                 (void) fprintf(stderr, "Don't know what to clear. "
219                     "Use -o or -m PATH with -x option.\n");
220             }
221
222         } else if (gflag) {
223             if (mflag) {
224                 if (fsd_get_param(&handle, mnt, &param) != 0) {
225                     ret = errout(&handle);
226                 } else {
227                     (void) printf("%s\n", mnt);
228                     print_fsd(&param);
229                 }
230             } else {
231                 (void) fprintf(stderr, "Don't know what to get. "
232                     "Use -m PATH with -g option.\n");
233             }
234
235         } else if (lflag) { /* add other disturbances here */
236             if (!(cflag && rflag)) {
237                 (void) fprintf(stderr, "Need chance and range.");
238                 goto end;
239             }
240
241             param.read_less_chance = chance;
242             param.read_less_r[0] = range[0];
243             param.read_less_r[1] = range[1];
244
245             if (oflag) {
246                 if (fsd_disturb_omni(&handle, &param) != 0)
247                     ret = errout(&handle);
248
249             } else if (mflag) {
250                 if (fsd_disturb(&handle, mnt, &param) != 0)
251                     ret = errout(&handle);
252
253             } else {
254                 (void) fprintf(stderr,
255                     "Don't know what to disturb. "
256                     "Use -o or -m PATH with this options.");
257             }
258
259         } else {

```

```
260 usage:
261         (void) fprintf(stderr, "Usage: fsdadm "
262             "[-ed] [-ai] [-o] [-x] [-g] [-l] "
263             "[-r range_start range_end]\n"
264             "\t[-c chance] [-m path]\n\n");
265
266         (void) fprintf(stderr,
267             "\t -e enable fsd\n"
268             "\t -d disable fsd\n"
269             "\t -a display disturbance parameters for all disturbed\n"
270             "\t     filesystems\n"
271             "\t -i display information about current fsd status\n"
272             "\t -o omnipresent switch\n"
273             "\t -x clear switch\n"
274             "\t -g get disturbance parameters\n"
275             "\t -l \"read less\" disturbance\n"
276             "\t     every read operation would read n (from a given\n"
277             "\t     range) bytes less than it was requested\n"
278             "\t -r range for some types of disturbances\n"
279             "\t -c chance of the disturbance\n"
280             "\t -m path to mountpoint (or a representative file)\n"
281             "\n");
282     }
283
284 end:
285     fsd_close(&handle);
286     return (ret);
287 }
```

```

*****
13888 Tue Sep 3 18:09:40 2013
new/usr/src/lib/Makefile
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #

22 # Copyright 2011 Nexenta Systems, Inc. All rights reserved.
23 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright (c) 2012 by Delphix. All rights reserved.
25 # Copyright (c) 2012, Joyent, Inc. All rights reserved.

27 include ../Makefile.master

29 # Note that libcurses installs commands along with its library.
30 # This is a minor bug which probably should be fixed.
31 # Note also that a few extra libraries are kept in cmd source.
32 #
33 # Certain libraries are linked with, hence depend on, other libraries.
34 #
35 # Although we have historically used .WAIT to express dependencies, it
36 # reduces the amount of parallelism and thus lengthens the time it
37 # takes to build the libraries. Thus, we now require that any new
38 # libraries explicitly call out their dependencies. Eventually, all
39 # the library dependencies will be called out explicitly. See
40 # "Library interdependencies" near the end of this file.
41 #
42 # Aside from explicit dependencies (and legacy .WAITs), all libraries
43 # are built in parallel.
44 #
45 .PARALLEL:

47 #
48 # The $(CLOSED_BUILD) additions to SUBDIRS & MSGSUBDIRS are unfortunate,
49 # but required due to the "dependencies" of using .WAIT to barrier the
50 # parallel dmake builds. once 4631488 has been fixed, they can be
51 # consolidated into one $(CLOSED_BUILD)SUBDIRS += (all closed libs) as
52 # shown in HDRSUBDIRS
53 #
54 SUBDIRS= \
55     common .WAIT \
56     ../cmd/sgs/libconv .WAIT \
57     ../cmd/sgs/libdl .WAIT

59 SUBDIRS += \
60     libc .WAIT \
61     ../cmd/sgs/libelf .WAIT

```

```

62     c_synonyms \
63     libmd \
64     libmd5 \
65     librm \
66     libmp .WAIT \
67     libnsl \
68     libsecdb .WAIT \
69     librpcsvc \
70     libsocket .WAIT \
71     libsctp \
72     libsip \
73     libcommputil \
74     libresolv \
75     libresolv2 .WAIT \
76     libw .WAIT \
77     libintl .WAIT \
78     ../cmd/sgs/librtld_db \
79     libaio \
80     libast \
81     libdll \
82     libcmd \
83     libshell \
84     libsum \
85     librt \
86     libadm \
87     libctf \
88     libdtrace \
89     libdtrace_jni \
90     libcurses \
91     libtermcap \
92     libgen \
93     libgss \
94     libpam \
95     libuuid \
96     libthread \
97     libpthreads .WAIT \
98     libslp \
99     libbsdmalloc \
100    libdoor \
101    libdevinfo \
102    libdladm \
103    libdlpi \
104    libeti \
105    libcrypt \
106    libdns_sd \
107    libefi \
108    libfstyp \
109    libwanboot \
110    libwanbootutil \
111    libcryptoutil \
112    libinetutil \
113    libipadm \
114    libipmp \
115    libiscsit \
116    libkmf \
117    libkstat \
118    libkvm \
119    liblm \
120    libmalloc \
121    libmapmalloc \
122    libmtmalloc \
123    libnls \
124    libnwam \
125    libsbios \
126    libtecla \
127    libumem \

```

## new/usr/src/lib/Makefile

3

```

128     libnvpair      .WAIT  \
129     libexacct      \
130     libsas1        \
131     libldap5       \
132     libslldap      .WAIT  \
133     libbsm         \
134     libsys         \
135     libsysevent    \
136     libnisdb       \
137     libpool        \
138     libpp          \
139     libproc        \
140     libproject     \
141     libsendfile    \
142     nametoaddr     \
143     ncad_addr      \
144     hbaapi         \
145     smhba          \
146     sun_fc         \
147     sun_sas        \
148     gss_mechs/mech_krb5 .WAIT  \
149     libkrb5       .WAIT  \
150     krb5           .WAIT  \
151     libsmvfs       \
152     libfcoe        \
153     libsrpt        \
154     libstmf        \
155     libstmfproxy   \
156     libnsctl       \
157     libunistat     \
158     libdscfg       \
159     librdc         \
160     libinstzones   \
161     libpkg         \
162     libpcidb      \
163     libfsd       \
162     libpcidb

```

```

165 SUBDIRS += \
166     passwdutil    \
167     pam_modules   \
168     crypt_modules \
169     libadt_jni    \
170     abi           \
171     auditd_plugins \
172     libvolmgt     \
173     libdevice     \
174     libdevvid     \
175     libdhcpsvc    \
176     libc_db       \
177     libndmp       \
178     libsec        \
179     libtnfprobe   \
180     libtnf        \
181     libtnfctl     \
182     libdhcpageant \
183     libdhcpcdu    \
184     libdhcputil   \
185     libxnet       \
186     libipsecutil  \
187 $(CLOSED_BUILD)SUBDIRS += \
188     $(CLOSED)/lib/libike
189 SUBDIRS += \
190     nswitch       \
191     print         \
192     libuutil      \

```

## new/usr/src/lib/Makefile

4

```

193     libscf         \
194     libinetsvc     \
195     librestart     \
196     libsched       \
197     libelfsign     \
198     pkcs11         .WAIT  \
199     libpctx        .WAIT  \
200     libcpc         \
201     watchmalloc    \
202     extendedFILE   \
203     madv           \
204     mpss           \
205     libdisasm      \
206     libwrap        \
207     libxcurses     \
208     libxcurses2    \
209     libbrand       .WAIT  \
210     libzonecfg     \
211     libzoneinfo    \
212     libzonestat    \
213     libtsnet       \
214     libtsol        \
215     gss_mechs/mech_spnego \
216     gss_mechs/mech_dummy \
217     gss_mechs/mech_dh   \
218     rpcsec_gss      \
219     libraidcfg     .WAIT  \
220     librcm         .WAIT  \
221     libcfgadm      .WAIT  \
222     libpicl        .WAIT  \
223     libpicltree    .WAIT  \
224     raidcfg_plugins \
225     cfgadm_plugins \
226     libmail        \
227     lvm            \
228     libsmmedia     \
229     libipp         \
230     libdiskmgt     \
231     liblgrp        \
232     libfsmgt       \
233     fm             \
234     libavl         \
235     libcmdutils    \
236     libcontract    \
237     ../cmd/sendmail/libmilter \
238     sasl_plugins   \
239     udapl          \
240     libzpool       \
241     libzfs_core    \
242     libzfs         \
243     libbe          \
244     pylibbe        \
245     libzfs_jni     \
246     pyzfs          \
247     pysolaris      \
248     libmapi        \
249     brand          \
250     policykit      \
251     hal            \
252     libshare       \
253     libsqlite      \
254     libidmap       \
255     libadutils     \
256     libipmi        \
257     libexacct/demo \
258     libvrrpadm     \

```

## new/usr/src/lib/Makefile

```

259     libvscan          \
260     libgrubmgmt      \
261     smbshr           \
262     libilb           \
263     scsi             \
264     libima           \
265     libsun_ima       \
266     mpapi            \
267     librstp          \
268     librepase        \
269     libhotplug       \
270     libfruutils     .WAIT \
271     libfru           \
272     $(MACH)_SUBDIRS)

274 i386_SUBDIRS=      \
275     libntfs         \
276     libparted       \
277     libfdisk        \
278     libsaveargs

280 sparc_SUBDIRS= .WAIT \
281     efcode          \
282     libds           \
283     libdscp         \
284     libprtdiag     .WAIT \
285     libprtdiag_psr \
286     libpri          \
287     librsc         \
288     storage         \
289     libpcp          \
290     libtsalarm      \
291     libvl2n

293 FM_sparc_DEPLIBS= libpri

295 fm:
296     libexacct       \
297     libipmi         \
298     libzfs          \
299     scsi            \
300     $(FM_$(MACH)_DEPLIBS)

302 #
303 # Create a special version of $(SUBDIRS) with no .WAIT's, for use with the
304 # clean and clobber targets (for more information, see those targets, below).
305 #
306 NOWAIT_SUBDIRS= $(SUBDIRS:.WAIT=)

308 DCSSUBDIRS =      \
309     lvm

311 MSGSUBDIRS=      \
312     abi             \
313     auditd_plugins \
314     brand           \
315     cfgadm_plugins \
316     gss_mechs/mech_dh \
317     gss_mechs/mech_krb5 \
318     krb5            \
319     libast         \
320     libbsm         \
321     libc           \
322     libcfgadm      \
323     libcmd         \
324     libcontract    \

```

5

## new/usr/src/lib/Makefile

```

325     libcurses       \
326     libdhcpsvc     \
327     libdhcputil    \
328     libipseutil    \
329     libdiskmgmt    \
330     libdladm       \
331     libdll         \
332     libgrubmgmt    \
333     libgss         \
334     libidmap       \
335     libipmp        \
336     libilb         \
337     libinetutil    \
338     libinstzones   \
339     libipadm       \
340     libnsl         \
341     libnwam        \
342     libpam         \
343     libpicl        \
344     libpool        \
345     libpkg         \
346     libpp          \
347     libscf         \
348     libsas1        \
349     libldap5       \
350     libsecdb       \
351     libshare       \
352     libshell       \
353     libslldap     \
354     libslp         \
355     libsmbfs       \
356     libsmmedia     \
357     libsum         \
358     libtsol        \
359     libuutil       \
360     libvrrpadm     \
361     libvscan       \
362     libwanboot     \
363     libwanbootutil \
364     libzfs         \
365     libzonecfg     \
366     lvm            \
367     madv           \
368     mpss           \
369     pam_modules    \
370     pyzfs          \
371     pysolaris      \
372     rpcsec_gss     \
373     librepase      \
374 MSGSUBDIRS += \
375     $(MACH)_MSGSUBDIRS)

377 sparc_MSGSUBDIRS= \
378     libprtdiag     \
379     libprtdiag_psr

381 i386_MSGSUBDIRS= libfdisk

383 HDRSUBDIRS=      \
384     auditd_plugins \
385     libast         \
386     libbrand       \
387     libbsm         \
388     libc           \
389     libcmd         \
390     libcmdutils    \

```

6

## new/usr/src/lib/Makefile

```

391 libcommputil \
392 libcontract \
393 libcpc \
394 libctf \
395 libcurses \
396 libtermcap \
397 libcryptoutil \
398 libdevice \
399 libdevide \
400 libdevinfo \
401 libdiskmgt \
402 libdladm \
403 libdll \
404 libdlpi \
405 libdhcpageant \
406 libdhcpcsvc \
407 libdhcputil \
408 libdisasm \
409 libdns_sd \
410 libdscfg \
411 libdtrace \
412 libdtrace_jni \
413 libelfsign \
414 libeti \
415 libfru \
416 libfstyp \
417 libgen \
418 libipadm \
419 libipsecutil \
420 libinetsvc \
421 libinetutil \
422 libinstzones \
423 libipmi \
424 libipmp \
425 libipp \
426 libiscsit \
427 libkstat \
428 libkvm \
429 libmail \
430 libmd \
431 libmtmalloc \
432 libndmp \
433 libnvpair \
434 libnsctl \
435 libnsl \
436 libnwam \
437 libpam \
438 libpcidb \
439 libpctx \
440 libpicl \
441 libpicltree \
442 libpool \
443 libpp \
444 libproc \
445 libraidcfg \
446 librcm \
447 librdc \
448 libscf \
449 libsip \
450 libsbios \
451 librestart \
452 librpcsvc \
453 librsm \
454 librstp \
455 libsas1 \
456 libsec \

```

7

## new/usr/src/lib/Makefile

```

457 libshell \
458 libslp \
459 libsmmedia \
460 libsocket \
461 libsqlite \
462 libfcoe \
463 libsrpt \
464 libstmf \
465 libstmfproxy \
466 libsum \
467 libsysevent \
468 libtecla \
469 libtnf \
470 libtnfctl \
471 libtnfprobe \
472 libtsnet \
473 libtsol \
474 libvrrpadm \
475 libvolmgt \
476 libumem \
477 libunistat \
478 libuutil \
479 libwanboot \
480 libwanbootutil \
481 libwrap \
482 libxcurses2 \
483 libzfs \
484 libzfs_core \
485 libzfs_jni \
486 libzoneinfo \
487 libzonestat \
488 hal \
489 policykit \
490 lvm \
491 pkcs11 \
492 passwdutil \
493 ../cmd/sendmail/libmilter \
494 fm \
495 udapl \
496 libmapid \
497 libkrb5 \
498 libsmbfs \
499 libshare \
500 libidmap \
501 libvsan \
502 libgrubmgt \
503 smbsrv \
504 libilb \
505 scsi \
506 hbaapi \
507 smhba \
508 libima \
509 libsun_ima \
510 mpapi \
511 librepase \
512 libfsd \
513 ${$(MACH)_HDRSUBDIRS}

515 ${CLOSED_BUILD}HDRSUBDIRS += \
516 ${CLOSED}/lib/libike

518 i386_HDRSUBDIRS= \
519 libparted \
520 libfdisk \
521 libsaveargs \

```

8

```

523 sparc_HDRSUBDIRS= \
524     libds \
525     libdscp \
526     libpri \
527     libv12n \
528     storage

530 all := TARGET= all
531 check := TARGET= check
532 clean := TARGET= clean
533 clobber := TARGET= clobber
534 install := TARGET= install
535 install_h := TARGET= install_h
536 lint := TARGET= lint
537 _dc := TARGET= _dc
538 _msg := TARGET= _msg

540 .KEEP_STATE:

542 #
543 # For the all and install targets, we clearly must respect library
544 # dependencies so that the libraries link correctly. However, for
545 # the remaining targets (check, clean, clobber, install_h, lint, _dc
546 # and _msg), libraries do not have any dependencies on one another
547 # and thus respecting dependencies just slows down the build.
548 # As such, for these rules, we use pattern replacement to explicitly
549 # avoid triggering the dependency information. Note that for clean,
550 # clobber and lint, we must use $(NOWAIT_SUBDIRS) rather than
551 # $(SUBDIRS), to prevent '.WAIT' from expanding to '.WAIT-nodepend'.
552 #

554 all: $(SUBDIRS)

556 install: $(SUBDIRS) .WAIT install_extra

558 # extra libraries kept in other source areas
559 install_extra:
560     @cd ../cmd/sgs; pwd; $(MAKE) install_lib
561     @pwd

563 clean clobber lint: $(NOWAIT_SUBDIRS:%=%-nodepend)

565 install_h check: $(HDRSUBDIRS:%=%-nodepend)

567 _msg: $(MSGSUBDIRS:%=%-nodepend) .WAIT _dc

569 _dc: $(DCSUBDIRS:%=%-nodepend)

571 #
572 # Library interdependencies are called out explicitly here
573 #
574 auditd_plugins: libbsm libnsl libsecdb
575 gss_mechs/mech_krb5: libgss libnsl libsocket libresolv pkcs11
576 libadt_jni: libbsm
577 libast: libsocket
578 libadutils: libldap5 libresolv libsocket libnsl
579 nsswitch: libadutils libidmap
580 libbe: libzfs
581 libbsm: libtsol
582 libcmd: libsum libast libsocket libnsl
583 libcmdutils: libavl
584 libcontract: libnvpair
585 libdevinfo: libdevinfo
586 libdevinfo: libnvpair libsec
587 libdhcpcagent: libsocket libdhcputil libuuid libdlpi libcontract
588 libdhcpsvc: libinetutil

```

```

589 libdhcputil: libnsl libgen libinetutil libdlpi
590 libdladm: libdevinfo libinetutil libsocket libscf librcm libnvpair \
591     libxacct libnsl libkstat libcurses
592 libdll: libast
593 libdlpi: libinetutil libdladm
594 libds: libsysevent
595 libdscfg: libnsctl libunistat libsocket libnsl
596 libdtrace: libproc libgen libctf
597 libdtrace_jni: libuutil libdtrace
598 libefi: libuuid
599 libfstyp: libnvpair
600 libelfsign: libcryptoutil libkmf
601 libidmap: libadutils libldap5 libavl libldap libuutil
602 libipadm: libnsl libinetutil libsocket libdlpi libnvpair libdhcpcagent \
603     libdladm libsecdb
604 libiscsit: libc libnvpair libstmf libuuid libnsl
605 libkmf: libcryptoutil pkcs11
606 libnsl: libmd5
607 libmapid: libresolv
608 librdc: libsocket libnsl libnsctl libunistat libdscfg
609 libuuid: libdlpi
610 $(CLOSED_BUILD)libike: libipsecutil libxnet libcryptoutil
611 libinetutil: libsocket
612 libipsecutil: libtecla libsocket
613 libinstzones: libzonecfg libcontract
614 libpkg: libwanboot libscf libadm
615 libnwm: libscf
616 libsecdb: libnsl
617 libsas1: libgss libsocket pkcs11 libmd
618 sasl_plugins: pkcs11 libgss libsocket libsas1
619 libscfp: libsocket
620 libshell: libast libcmd libdll libsocket libsecdb
621 libsip: libmd5
622 libsmbsfs: libcmdutils libsocket libnsl libkrb5
623 libsocket: libnsl
624 libstmfproxy: libstmf libsocket libnsl libpthread
625 libsum: libast
626 libsysevent: libsecdb
627 libldap5: libsas1 libsocket libnsl libmd
628 libldap: libldap5 libtsol libnsl libc libscf libresolv
629 libpool: libnvpair libxacct
630 libpp: libast
631 libzonecfg: libc libsocket libnsl libuuid libnvpair libsysevent libsec \
632     libbrand libpool libscf
633 libproc: ../cmd/sgs/librtld_db ../cmd/sgs/libelf libctf libsaveargs
634 libproject: libpool libproc libsecdb
635 libtermcap: libcurses
636 libtsnet: libnsl libtsol libsecdb
637 libwrap: libnsl libsocket
638 libwanboot: libnvpair libresolv libnsl libsocket libdevinfo libinetutil \
639     libdhcputil
640 libwanbootutil: libnsl
641 pam_modules: libproject passwdutil smbsrv
642 libscf: libuutil libmd libgen libsmbios libnsl
643 libnetsvc: libscf
644 librestart: libuutil libscf
645 libsaveargs: libdisasm
646 ../cmd/sgs/libdl: ../cmd/sgs/libconv
647 ../cmd/sgs/libelf: ../cmd/sgs/libconv
648 pkcs11: libcryptoutil
649 print: libldap5
650 udapl/udapl_tavor: udapl/libdat
651 libzfs: libdevinfo libgen libnvpair libuutil \
652     libadm libavl libefi libidmap libmd libzfs_core
653 libzfs_core: libnvpair
654 libzfs_jni: libdiskmgt libnvpair libzfs

```



```
655 libzpool:      libavl libumem libnvpair libcmdutils
656 libsec:        libavl libidmap
657 brand:         libc libsocket
658 libshare:      libscf libzfs libuuid libfsmgmt libsecdb libumem libsmfbs
659 libexacct/demo: libexacct libproject libsocket libnsl
660 libtsalarm:    libpcp
661 smbsrv:        libsocket libnsl libmd libxnet libpthread librt \
662                libshare libidmap pkcs11 libsqlite libcryptoutil \
663                libreparse libcmdutils
664 libv12n:       libds libuuid
665 libvrrpadm:    libsocket libdladm libscf
666 libvscan:      libscf
667 libfru:        libfruutils
668 scsi:          libnvpair libfru
669 mpapi:         libpthread libdevinfo libsysevent libnvpair
670 sun_fc:        libdevinfo libsysevent libnvpair
671 libsun_ima:    libdevinfo libsysevent libnsl
672 sun_sas:       libdevinfo libsysevent libnvpair libkstat libdevid
673 libgrubmgmt:  libdevinfo libzfs libfstyp
674 pylibbe:      libbe libzfs
675 pyzfs:        libnvpair libzfs
676 pysolaris:    libsec libidmap
677 libreparse:    libnvpair
678 libhotplug:   libnvpair
679 cfgadm_plugins: libhotplug
680 libilb:       libsocket
681 $(INTEL_BUILD)libdiskmgmt:libfdisk

683 #
684 # The reason this rule checks for the existence of the
685 # Makefile is that some of the directories do not exist
686 # in certain situations (e.g., exportable source builds,
687 # OpenSolaris).
688 #
689 $(SUBDIRS): FRC
690     @if [ -f $@/Makefile ]; then \
691         cd $@; pwd; $(MAKE) $(TARGET); \
692     else \
693         true; \
694     fi

696 $(SUBDIRS:%=%-nodepend):
697     @if [ -f $@:%-nodepend=%/Makefile ]; then \
698         cd $@:%-nodepend=%; pwd; $(MAKE) $(TARGET); \
699     else \
700         true; \
701     fi

703 FRC:
```

new/usr/src/lib/libfsd/Makefile

1

```
*****
1361 Tue Sep  3 18:09:40 2013
new/usr/src/lib/libfsd/Makefile
fsh, fsd, libfsd, fsdadm
*****
```

```
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 # Copyright 2006 Sun Microsystems, Inc. All rights reserved.
22 # Use is subject to license terms.
23 #
24 # Copyright 2013 Damian Bogel. All rights reserved.
25 #

27 include ../Makefile.lib

29 HDRS = libfsd.h
30 HDRDIR = common
31 SUBDIRS = $(MACH)
32 $(BUILD64)SUBDIRS += $(MACH64)

34 all := TARGET= all
35 clean := TARGET= clean
36 clobber := TARGET= clobber
37 install := TARGET= install
38 lint := TARGET= lint

40 .KEEP_STATE:

42 all clean clobber install lint: $(SUBDIRS)

44 install_h: $(ROOTHDRS)

46 check: $(CHECKHDRS)

48 $(SUBDIRS): FRC
49 @cd $@; pwd; $(MAKE) $(TARGET)

51 FRC:

53 include ../Makefile.targ
```

new/usr/src/lib/libfsd/Makefile.com

1

```
*****
1261 Tue Sep  3 18:09:40 2013
new/usr/src/lib/libfsd/Makefile.com
fsh, fsd, libfsd, fsdadm
*****
```

```
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
22 # Use is subject to license terms.
23 #
24 # Copyright 2013 Damian Bogel. All rights reserved.
25 #

27 LIBRARY = libfsd.a
28 VERS    = .1
29 OBJECTS = libfsd.o

31 include ../../Makefile.lib

33 SRCDIR =      ../common
34 LIBS =      $(DYNLIB) $(LINTLIB)
35 $(LINTLIB) := SRCS = $(SRCDIR)/$(LINTSRC)
36 LDLIBS +=    -lc

38 CFLAGS +=    $(CCVERBOSE)
39 CPPFLAGS +=  -I../common

41 .KEEP_STATE:

43 all:        $(LIBS)

45 lint:       lintcheck

47 include ../../Makefile.targ
```

new/usr/src/lib/libfsd/amd64/Makefile

1

\*\*\*\*\*

557 Tue Sep 3 18:09:40 2013

new/usr/src/lib/libfsd/amd64/Makefile

fsh, fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 #
11 #
12 #
13 # Copyright 2013 Damian Bogel. All rights reserved.
14 #
15 include ../Makefile.com
16 include ../../Makefile.lib.64
17 #
18 install: all $(ROOTLIBS64) $(ROOTLINKS64)
```

```

*****
      8274 Tue Sep  3 18:09:40 2013
new/usr/src/lib/libfsd/common/libfsd.c
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel.  All rights reserved.
14  */

16 #include <fcntl.h>
17 #include <libfsd.h>
18 #include <string.h>
19 #include <stropts.h>
20 #include <sys/debug.h>
21 #include <sys/errno.h>
22 #include <sys/fsd.h>
23 #include <sys/stat.h>
24 #include <sys/types.h>
25 #include <unistd.h>

28 /*
29  * libfsd
30  * Library used to drive the fsd pseudo-device driver.
31  *
32  * 1. Usage.
33  * fsd_handle_t is used for every operation on the fsd driver. It is acquired by
34  * a call to fsd_open(). Every handle must be released by calling fsd_close().
35  *
36  * A handle is a structure that contains data needed to drive the fsd and get
37  * information about errors.
38  *
39  *
40  * Basics:
41  * A disturber is a hook for filesystem operations. There are different types of
42  * disturbers, but the property that connects them is that EVERY program that
43  * uses filesystem API should expect that kind of behaviour. Doing otherwise is
44  * a bug and could lead to serious problems.
45  * Omnipresent disturber is a hook that is being installed whenever a new vfs_t
46  * is mounted.
47  *
48  *
49  * Errors:
50  * In almost all the functions (except fsd_close()) return value equal to -1
51  * indicates that there was an error. Last error data is contained in the
52  * handle in two fields: fsd_errno and errno.
53  * fsd_errno could be one of EFSD_XXX. errno is nonzero if and only if fsd_errno
54  * is set to EFSD_CANT_OPEN_DRIVER or EFSD_CANT_OPEN_MOUNTPOINT.
55  * fsd_strerror() is used to get the error message from fsd_errno. errno should be
56  * treated according to Intro(2).
57  *
58  * Handle management:
59  * fsd_open()
60  * Returns a handle.
61  *

```

```

62  * fsd_close(handle)
63  * Destroys a handle.
64  *
65  *
66  * Enabling/disabling fsd:
67  * fsd_enable(handle)
68  * fsd_disable(handle)
69  * When fsd is enabled, it cannot be detached from the system. Otherwise,
70  * it could be and all the information that the fsd contains could be lost.
71  * The client should keep the fsd enabled whenever he needs to. No
72  * operations on fsd could be made if it's not enabled.
73  *
74  *
75  * Getting information:
76  * fsd_get_info(handle, info)
77  * Used to retrieve information such as:
78  * * whether fsd is enabled (system-wide)
79  * * whether there is an omnipresent disturber installed
80  * * omnipresent disturber's parameters
81  * * count of disturbers installed
82  *
83  * fsd_get_list(handle, fslist, count)
84  * Gets a list of disturbers installed. count is the maximum count of
85  * entries that could be returned by the function to the user-allocated
86  * buffer. count is changed to min(count, number of disturbers installed).
87  * There is no error returned if the number of disturbers installed exceeds
88  * the user-provided count. It just gets the maximal allowed amount of
89  * informaton.
90  *
91  * fsd_get_param(handle, path, param)
92  * Gets disturber parameters from a filesystem of the file pointed by the
93  * path.
94  *
95  *
96  * Installing/removing disturbers:
97  * fsd_disturb(handle, path, param)
98  * Installs (or changes, if a disturber on this filesystem already exists)
99  * a disturber on a filesystem of the file pointed by the path.
100 *
101 * fsd_disturb_omni(handle, param)
102 * Installs (or changes, if an omnipresent disturber already exists) an
103 * omnipresent disturber. Only one omnipresent disturber exists at a time.
104 *
105 * fsd_disturb_off(handle, path)
106 * fsd_disturb_omni_off(handle)
107 * Removes a disturber. It is guaranteed that after this function returns,
108 * the disturber won't change the behaviour of filesystem operations.
109 *
110 * 2. Multithreading.
111 * It is safe to use the libfsd API concurrently.
112 *
113 * Error information is encapsulated in a handle, so it's the client's job to
114 * properly share the handle between threads to preserve the consistency of
115 * error data within a handle.
116 */

118 extern int errno;

120 const char *
121 fsd_strerror(int e)
122 {
123     switch (e) {
124     case EFSD_NOERROR:
125         return ("no error");
126     case EFSD_BAD_PARAM:
127         return ("incorrect disturbance parameters");

```

```

128     case EFSD_INTERNAL:
129         return ("internal library error");
130     case EFSD_NOT_ENABLED:
131         return ("fsd is not enabled");
132     case EFSD_CANT_OPEN_DRIVER:
133         return ("cannot open fsd device");
134     case EFSD_CANT_OPEN_MOUNTPOINT:
135         return ("cannot open mountpoint");
136     case EFSD_ENTRY_NOT_FOUND:
137         return ("this filesystem is not being disturbed");
138     case EFSD_FAULT:
139         return ("bad pointer");
140     case EFSD_TOO_MANY_HOOKS:
141         return ("too many hooks");
142     case EFSD_UNKNOWN_ERROR:
143     default:
144         return ("unknown error");
145     }
146 }

148 static int
149 xlate_errno(int e)
150 {
151     switch (e) {
152     case 0:
153         return (0);
154     case (-1):
155         switch (errno) {
156         case 0:
157             return (EFSD_NOERROR);
158         case EFAULT:
159             return (EFSD_FAULT);
160         case ENOTTY:
161             return (EFSD_INTERNAL);
162         default:
163             return (EFSD_UNKNOWN_ERROR);
164         }
165     case ENOTACTIVE:
166         return (EFSD_NOT_ENABLED);
167     case ENOENT:
168         return (EFSD_ENTRY_NOT_FOUND);
169     case EINVAL:
170         return (EFSD_BAD_PARAM);
171     case EBADFD:
172         return (EFSD_INTERNAL);
173     case EAGAIN:
174         return (EFSD_TOO_MANY_HOOKS);
175     default:
176         return (EFSD_UNKNOWN_ERROR);
177     }
178 }

180 static int
181 ioctl_set_handle(fsd_handle_t *handle, int ioctlret)
182 {
183     handle->fsd_errno = xlate_errno(ioctlret);
184     handle->errno = 0;

186     if (handle->fsd_errno == 0)
187         return (0);
188     else
189         return (-1);
190 }

192 int
193 fsd_open(fsd_handle_t *handle)

```

```

194 {
195     if ((handle->fd = open(FSD_DEV_PATH, O_RDWR)) == -1) {
196         handle->fsd_errno = EFSD_CANT_OPEN_DRIVER;
197         handle->errno = errno;
198         return (-1);
199     }
200     return (0);
201 }

203 void
204 fsd_close(fsd_handle_t *handle)
205 {
206     (void) close(handle->fd);
207     handle->fd = -1;
208 }

211 int
212 fsd_enable(fsd_handle_t *handle)
213 {
214     return (ioctl_set_handle(handle, ioctl(handle->fd, FSD_ENABLE)));
215 }

216 }

218 int
219 fsd_disable(fsd_handle_t *handle)
220 {
221     return (ioctl_set_handle(handle, ioctl(handle->fd, FSD_DISABLE)));
222 }

224 int
225 fsd_disturb(fsd_handle_t *handle, const char *mnt_path, fsd_t *param)
226 {
227     fsd_ioc_t ioc;
228     int error;

230     (void) memcpy(&ioc.fsd_ioc_dis.fsdd_param, param,
231                 sizeof(ioc.fsd_ioc_dis.fsdd_param));

233     if ((ioc.fsd_ioc_dis.fsdd_mnt = open(mnt_path, O_RDONLY)) == -1) {
234         handle->fsd_errno = EFSD_CANT_OPEN_MOUNTPOINT;
235         handle->errno = errno;
236         return (-1);
237     }

239     error = ioctl(handle->fd, FSD_DISTURB, &ioc);
240     (void) close(ioc.fsd_ioc_dis.fsdd_mnt);
241     return (ioctl_set_handle(handle, error));
242 }

244 int
245 fsd_disturb_off(fsd_handle_t *handle, const char *mnt_path)
246 {
247     fsd_ioc_t ioc;
248     int error;

250     if ((ioc.fsd_ioc_mnt = open(mnt_path, O_RDONLY)) == -1) {
251         handle->fsd_errno = EFSD_CANT_OPEN_MOUNTPOINT;
252         handle->errno = errno;
253         return (-1);
254     }

256     error = ioctl(handle->fd, FSD_DISTURB_OFF, &ioc);
257     (void) close(ioc.fsd_ioc_mnt);
258     return (ioctl_set_handle(handle, error));
259 }

```

```

261 int
262 fsd_disturb_omni(fsd_handle_t *handle, fsd_t *param)
263 {
264     fsd_ioc_t ioc;
265
266     (void) memcpy(&ioc.fsd_ioc_param, param, sizeof (ioc.fsd_ioc_param));
267
268     return (ioctl_set_handle(handle, ioctl(handle->fd, FSD_DISTURB_OMNI,
269         &ioc)));
270 }
271
272 int
273 fsd_disturb_omni_off(fsd_handle_t *handle)
274 {
275     return (ioctl_set_handle(handle, ioctl(handle->fd,
276         FSD_DISTURB_OMNI_OFF)));
277 }
278
279 int
280 fsd_get_param(fsd_handle_t *handle, const char *mnt_path, fsd_t *param)
281 {
282     fsd_ioc_t ioc;
283     int error;
284     int mntfd;
285
286     if ((ioc.fsd_ioc_mnt = open(mnt_path, O_RDONLY)) == -1) {
287         handle->fsd_errno = EFSM_CANT_OPEN_MOUNTPOINT;
288         handle->errno = errno;
289         return (-1);
290     }
291     mntfd = ioc.fsd_ioc_mnt;
292
293     error = ioctl(handle->fd, FSD_GET_PARAM, &ioc);
294     if (error == 0)
295         (void) memcpy(param, &ioc.fsd_ioc_param, sizeof (*param));
296
297     (void) close(mntfd);
298
299     return (ioctl_set_handle(handle, error));
300 }
301
302 int
303 fsd_get_info(fsd_handle_t *handle, fsd_info_t *info)
304 {
305     fsd_ioc_t ioc;
306     int error;
307
308     error = ioctl(handle->fd, FSD_GET_INFO, &ioc);
309     if (error == 0)
310         (void) memcpy(info, &ioc.fsd_ioc_info, sizeof (*info));
311
312     return (ioctl_set_handle(handle, error));
313 }
314
315 int
316 fsd_get_list(fsd_handle_t *handle, fsd_fs_t *fslist, int *count)
317 {
318     fsd_ioc_t ioc;
319     int error;
320
321     ioc.fsd_ioc_list.listp = (uintptr_t)fslist;
322     ioc.fsd_ioc_list.count = *count;
323
324     error = ioctl(handle->fd, FSD_GET_LIST, &ioc);
325     *count = ioc.fsd_ioc_list.count;

```

```

326     return (ioctl_set_handle(handle, error));
327 }

```

new/usr/src/lib/libfsd/common/libfsd.h

1

```
*****
1728 Tue Sep 3 18:09:40 2013
new/usr/src/lib/libfsd/common/libfsd.h
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel. All rights reserved.
14 */

16 #ifndef _LIBFSD_H
17 #define _LIBFSD_H

19 #include <sys/fsd.h>

21 #ifdef __cplusplus
22 extern "C" {
23 #endif

25 typedef struct fsd_handle {
26     int fd;
27     int fsd_errno;
28     int errno;
29 } fsd_handle_t;

31 #define EFSD_NOERROR                0
32 #define EFSD_BAD_PARAM              1
33 #define EFSD_CANT_OPEN_DRIVER      2
34 #define EFSD_CANT_OPEN_MOUNTPOINT  3
35 #define EFSD_ENTRY_NOT_FOUND       4
36 #define EFSD_FAULT                  5
37 #define EFSD_NOT_ENABLED            6
38 #define EFSD_TOO_MANY_HOOKS        7
39 #define EFSD_INTERNAL               8
40 #define EFSD_UNKNOWN_ERROR          9

42 extern const char *fsd_strerror(int e);

44 extern int fsd_open(fsd_handle_t *handle);
45 extern void fsd_close(fsd_handle_t *handle);

47 extern int fsd_enable(fsd_handle_t *handle);
48 extern int fsd_disable(fsd_handle_t *handle);

50 extern int fsd_get_param(fsd_handle_t *handle, const char *mnt_path,
51                         fsd_t *param);
52 extern int fsd_disturb(fsd_handle_t *handle, const char *mnt_path,
53                       fsd_t *param);
54 extern int fsd_disturb_off(fsd_handle_t *handle, const char *mnt_path);

56 extern int fsd_disturb_omni(fsd_handle_t *handle, fsd_t *param);
57 extern int fsd_disturb_omni_off(fsd_handle_t *handle);

60 extern int fsd_get_info(fsd_handle_t *handle, fsd_info_t *info);
61 extern int fsd_get_list(fsd_handle_t *handle, fsd_fs_t *fslist, int *count);
```

new/usr/src/lib/libfsd/common/libfsd.h

2

```
63 #ifdef __cplusplus
64 }
65 #endif

67 #endif /* _LIBFSD_H */
```



new/usr/src/lib/libfsd/common/l1ib-lfsd

1

\*\*\*\*\*

531 Tue Sep 3 18:09:40 2013

new/usr/src/lib/libfsd/common/l1ib-lfsd

fsh, fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 Damian Bogel. All rights reserved.
14 */

16 /* LINTLIBRARY */
17 /* PROTLIB1 */

19 #include <libfsd.h>
```

new/usr/src/lib/libfsd/common/mapfile-vers

1

\*\*\*\*\*

1082 Tue Sep 3 18:09:40 2013

new/usr/src/lib/libfsd/common/mapfile-vers

fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 #
11 #
12 #
13 # Copyright 2013 Damian Bogel. All rights reserved.
14 #
15 #
16 #
17 # MAPFILE HEADER START
18 #
19 # WARNING: STOP NOW. DO NOT MODIFY THIS FILE.
20 # Object versioning must comply with the rules detailed in
21 #
22 #     usr/src/lib/README.mapfiles
23 #
24 # You should not be making modifications here until you've read the most current
25 # copy of that file. If you need help, contact a gatekeeper for guidance.
26 #
27 # MAPFILE HEADER END
28 #
29 #
30 $mapfile_version 2
31 #
32 SYMBOL_VERSION ILLUMOSprivate {
33     global:
34         fsd_close;
35         fsd_disable;
36         fsd_disturb;
37         fsd_disturb_off;
38         fsd_enable;
39         fsd_get_param;
40         fsd_disturb_omni;
41         fsd_disturb_omni_off;
42         fsd_open;
43         fsd_get_list;
44         fsd_get_info;
45         fsd_strerror;
46     local:
47         *;
48 };
```

new/usr/src/lib/libfsd/i386/Makefile

1

\*\*\*\*\*

523 Tue Sep 3 18:09:40 2013

new/usr/src/lib/libfsd/i386/Makefile

fsh, fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 #
11 #
12 #
13 # Copyright 2013 Damian Bogel. All rights reserved.
14 #
15 include ../Makefile.com
16 #
17 install: all $(ROOTLIBS) $(ROOTLINKS)
```

```

*****
43444 Tue Sep 3 18:09:40 2013
new/usr/src/uts/common/Makefile.files
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 #
22 #
23 # Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright (c) 2012 Nexenta Systems, Inc. All rights reserved.
25 # Copyright (c) 2012 by Delphix. All rights reserved.
26 # Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
27 #
28 #
29 #
30 # This Makefile defines all file modules for the directory uts/common
31 # and its children. These are the source files which may be considered
32 # common to all SunOS systems.
33 #
34 i386_CORE_OBJS += \
35     atomic.o          \
36     avintr.o          \
37     pic.o
38 #
39 sparc_CORE_OBJS +=
40 #
41 COMMON_CORE_OBJS += \
42     beep.o            \
43     bitset.o          \
44     bp_map.o          \
45     brand.o           \
46     cpucaps.o         \
47     cmt.o             \
48     cmt_policy.o     \
49     cpu.o             \
50     cpu_event.o       \
51     cpu_intr.o        \
52     cpu_pm.o          \
53     cpupart.o         \
54     cap_util.o        \
55     disp.o            \
56     group.o           \
57     kstat_fr.o        \
58     iscsiboot_prop.o \
59     lgrp.o             \
60     lgrp_topo.o       \
61     mmapobj.o

```

```

62     mutex.o           \
63     page_lock.o       \
64     page_retire.o     \
65     panic.o           \
66     param.o           \
67     pg.o              \
68     pghw.o            \
69     putnext.o         \
70     rctl_proc.o       \
71     rwlock.o          \
72     seg_kmem.o        \
73     softint.o         \
74     string.o          \
75     strtol.o          \
76     strtoul.o         \
77     strtoll.o         \
78     strtoull.o        \
79     thread_intr.o    \
80     vm_page.o         \
81     vm_pagelist.o     \
82     zlib_obj.o        \
83     clock_tick.o
84 #
85 CORE_OBJS += $(COMMON_CORE_OBJS) $(MACH)_CORE_OBJS
86 #
87 ZLIB_OBJS = zutil.o zmod.o zmod_subr.o \
88     adler32.o crc32.o deflate.o inffast.o \
89     inflate.o inftrees.o trees.o
90 #
91 GENUNIX_OBJS += \
92     access.o          \
93     acl.o             \
94     acl_common.o      \
95     adjtime.o         \
96     alarm.o           \
97     aio_subr.o        \
98     auditsys.o        \
99     audit_core.o      \
100    audit_zone.o       \
101    audit_memory.o     \
102    autoconf.o         \
103    avl.o              \
104    bdev_dsort.o       \
105    bio.o              \
106    bitmap.o           \
107    blabel.o           \
108    brandsys.o         \
109    bz2blocksort.o     \
110    bz2compress.o      \
111    bz2decompress.o    \
112    bz2randtable.o    \
113    bz2bzlib.o         \
114    bz2crctable.o     \
115    bz2huffman.o       \
116    callb.o            \
117    callout.o          \
118    chdir.o            \
119    chmod.o            \
120    chown.o            \
121    cladm.o            \
122    class.o            \
123    clock.o            \
124    clock_highres.o    \
125    clock_realtime.o  \
126    close.o            \
127    compress.o

```

## new/usr/src/uts/common/Makefile.files

```

128 condvar.o \
129 conf.o \
130 console.o \
131 contract.o \
132 copyops.o \
133 core.o \
134 corectl.o \
135 cred.o \
136 cs_stubs.o \
137 dacf.o \
138 dacf_clnt.o \
139 damap.o \
140 cyclic.o \
141 ddi.o \
142 ddifm.o \
143 ddi_hp_impl.o \
144 ddi_hp_ndi.o \
145 ddi_intr.o \
146 ddi_intr_impl.o \
147 ddi_intr_irm.o \
148 ddi_nodeid.o \
149 ddi_timer.o \
150 devcfg.o \
151 devcache.o \
152 device.o \
153 devid.o \
154 devid_cache.o \
155 devid_scsi.o \
156 devid_smp.o \
157 devpolicy.o \
158 disp_lock.o \
159 dnlc.o \
160 driver.o \
161 dumpsubr.o \
162 driver_lyr.o \
163 dtrace_subr.o \
164 errorq.o \
165 etheraddr.o \
166 evchannels.o \
167 exacct.o \
168 exacct_core.o \
169 exec.o \
170 exit.o \
171 fbio.o \
172 fcntl.o \
173 fdbuffer.o \
174 fdsync.o \
175 fem.o \
176 ffs.o \
177 fio.o \
178 flock.o \
179 fm.o \
180 fork.o \
181 fsh.o \
182 vpm.o \
183 fs_reparse.o \
184 fs_subr.o \
185 fsflush.o \
186 ftrace.o \
187 getcwd.o \
188 getdents.o \
189 getloadavg.o \
190 getpagesizes.o \
191 getpid.o \
192 gfs.o \
193 rusagesys.o \

```

3

## new/usr/src/uts/common/Makefile.files

```

194 gid.o \
195 groups.o \
196 grow.o \
197 hat_refmod.o \
198 id32.o \
199 id_space.o \
200 inet_ntop.o \
201 instance.o \
202 ioctl.o \
203 ip_cksum.o \
204 issetugid.o \
205 ippconf.o \
206 kpcp.o \
207 kdi.o \
208 kiconv.o \
209 klpd.o \
210 kmem.o \
211 ksyms_snapshot.o \
212 l_strplumb.o \
213 labelsys.o \
214 link.o \
215 list.o \
216 lockstat_subr.o \
217 log_sysevent.o \
218 logsubr.o \
219 lookup.o \
220 lseek.o \
221 ltos.o \
222 lwp.o \
223 lwp_create.o \
224 lwp_info.o \
225 lwp_self.o \
226 lwp_obj.o \
227 lwp_timer.o \
228 lwpsys.o \
229 main.o \
230 mmapobjsys.o \
231 memcntl.o \
232 memstr.o \
233 lgrpsys.o \
234 mkdir.o \
235 mknod.o \
236 mount.o \
237 move.o \
238 msacct.o \
239 multidata.o \
240 nbmlock.o \
241 ndifm.o \
242 nice.o \
243 netstack.o \
244 ntptime.o \
245 nvpair.o \
246 nvpair_alloc_system.o \
247 nvpair_alloc_fixed.o \
248 fnvpair.o \
249 octet.o \
250 open.o \
251 p_online.o \
252 pathconf.o \
253 pathname.o \
254 pause.o \
255 serializer.o \
256 pci_intr_lib.o \
257 pci_cap.o \
258 pcifm.o \
259 pgrp.o \

```

4

## new/usr/src/uts/common/Makefile.files

```

260      pgrpsys.o  \
261      pid.o      \
262      pkp_hash.o \
263      policy.o   \
264      poll.o     \
265      pool.o     \
266      pool_pset.o \
267      port_subr.o \
268      ppriv.o    \
269      printf.o   \
270      priocntl.o \
271      priv.o     \
272      priv_const.o \
273      proc.o     \
274      procset.o  \
275      processor_bind.o \
276      processor_info.o \
277      profil.o   \
278      project.o  \
279      qsort.o    \
280      rctl.o     \
281      rctlsys.o \
282      readlink.o \
283      refstr.o   \
284      rename.o   \
285      resolvepath.o \
286      retire_store.o \
287      process.o  \
288      rlimit.o   \
289      rmap.o     \
290      rw.o       \
291      rwstlock.o \
292      sad_conf.o \
293      sid.o      \
294      sidsys.o   \
295      sched.o    \
296      schedctl.o \
297      sctp_crc32.o \
298      seg_dev.o  \
299      seg_kp.o   \
300      seg_kpm.o  \
301      seg_map.o  \
302      seg_vn.o   \
303      seg_spt.o  \
304      semaphore.o \
305      sendfile.o \
306      session.o  \
307      share.o    \
308      shuttle.o  \
309      sig.o      \
310      sigaction.o \
311      sigaltstack.o \
312      signotify.o \
313      sigpending.o \
314      sigprocmask.o \
315      sigqueue.o \
316      sigsendset.o \
317      sigsuspend.o \
318      sigtimedwait.o \
319      sleepq.o   \
320      sock_conf.o \
321      space.o    \
322      sscanf.o   \
323      stat.o     \
324      statfs.o   \
325      statvfs.o  \

```

5

## new/usr/src/uts/common/Makefile.files

```

326      stol.o     \
327      str_conf.o \
328      strcalls.o \
329      stream.o   \
330      streamio.o \
331      strext.o   \
332      strsubr.o  \
333      strsun.o   \
334      subr.o     \
335      sunddi.o   \
336      sunmdi.o   \
337      sunndi.o   \
338      sunpci.o   \
339      sunpm.o    \
340      sundlpi.o  \
341      suntpi.o   \
342      swap_subr.o \
343      swap_vnops.o \
344      symlink.o  \
345      sync.o     \
346      sysclass.o \
347      sysconfig.o \
348      sysent.o   \
349      sysfs.o    \
350      systeminfo.o \
351      task.o     \
352      taskq.o    \
353      tasksys.o  \
354      time.o     \
355      timer.o    \
356      times.o    \
357      timers.o   \
358      thread.o   \
359      tlabel.o   \
360      tnf_res.o  \
361      turnstile.o \
362      tty_common.o \
363      u8_textprep.o \
364      uadmin.o   \
365      uconv.o    \
366      ucredsys.o \
367      uid.o      \
368      umask.o    \
369      umount.o   \
370      uname.o    \
371      unix_bb.o  \
372      unlink.o   \
373      urw.o      \
374      utime.o    \
375      utssys.o   \
376      uucopy.o   \
377      vfs.o      \
378      vfs_conf.o \
379      vmem.o     \
380      vm_anon.o  \
381      vm_as.o    \
382      vm_meter.o \
383      vm_pageout.o \
384      vm_pvn.o   \
385      vm_rm.o    \
386      vm_seg.o   \
387      vm_subr.o  \
388      vm_swap.o  \
389      vm_usage.o \
390      vnode.o    \
391      vuid_queue.o \

```

6

## new/usr/src/uts/common/Makefile.files

7

```

392          vuid_store.o  \
393          waitq.o       \
394          watchpoint.o  \
395          yield.o       \
396          scsi_confdata.o \
397          xattr.o       \
398          xattr_common.o \
399          xdr_mblk.o    \
400          xdr_mem.o     \
401          xdr.o         \
402          xdr_array.o   \
403          xdr_refer.o   \
404          xhat.o        \
405          zone.o

407 #
408 #     Stubs for the stand-alone linker/loader
409 #
410 sparc_GENSTUBS_OBJS = \
411     kobj_stubs.o

413 i386_GENSTUBS_OBJS =

415 COMMON_GENSTUBS_OBJS =

417 GENSTUBS_OBJS += $(COMMON_GENSTUBS_OBJS) $($ (MACH)_GENSTUBS_OBJS)

419 #
420 #     DTrace and DTrace Providers
421 #
422 DTRACE_OBJS += dtrace.o dtrace_isa.o dtrace_asm.o

424 SDT_OBJS += sdt_subr.o

426 PROFILE_OBJS += profile.o

428 SYSTRACE_OBJS += systrace.o

430 LOCKSTAT_OBJS += lockstat.o

432 FASTTRAP_OBJS += fasttrap.o fasttrap_isa.o

434 DCPC_OBJS += dcpc.o

436 #
437 #     Driver (pseudo-driver) Modules
438 #
439 IPP_OBJS += ippctl.o

441 AUDIO_OBJS += audio_client.o audio_ddi.o audio_engine.o \
442     audio_fltdata.o audio_format.o audio_ctrl.o \
443     audio_grc3.o audio_output.o audio_input.o \
444     audio_oss.o audio_sun.o

446 AUDIOEMU10K_OBJS += audioemu10k.o

448 AUDIOENS_OBJS += audioens.o

450 AUDIOVIA823X_OBJS += audiovia823x.o

452 AUDIOVIA97_OBJS += audiovia97.o

454 AUDIO1575_OBJS += audio1575.o

456 AUDIO810_OBJS += audio810.o

```

## new/usr/src/uts/common/Makefile.files

8

```

458 AUDIOCMI_OBJS += audiocmi.o

460 AUDIOCMIHD_OBJS += audiocmihd.o

462 AUDIOHD_OBJS += audiohd.o

464 AUDIOIXP_OBJS += audioixp.o

466 AUDIOLS_OBJS += audiols.o

468 AUDIOP16X_OBJS += audiop16x.o

470 AUDIOPCI_OBJS += audiopci.o

472 AUDIOSOLO_OBJS += audiosolo.o

474 AUDIOTS_OBJS += audiots.o

476 AC97_OBJS += ac97.o ac97_ad.o ac97_alc.o ac97_cmi.o

478 BLKDEV_OBJS += blkdev.o

480 CARDBUS_OBJS += cardbus.o cardbus_hp.o cardbus_cfg.o

482 CONSKBD_OBJS += conskbd.o

484 CONSMS_OBJS += consms.o

486 OLDPTY_OBJS += tty_ptyconf.o

488 PTC_OBJS += tty_pty.o

490 PTSL_OBJS += tty_pts.o

492 PTM_OBJS += ptm.o

494 MII_OBJS += mii.o mii_cicada.o mii_natsemi.o mii_intel.o mii_qualsemi.o \
495     mii_marvell.o mii_realtek.o mii_other.o

497 PTS_OBJS += pts.o

499 PTY_OBJS += ptms_conf.o

501 SAD_OBJS += sad.o

503 MD4_OBJS += md4.o md4_mod.o

505 MD5_OBJS += md5.o md5_mod.o

507 SHA1_OBJS += sha1.o sha1_mod.o

509 SHA2_OBJS += sha2.o sha2_mod.o

511 IPGPC_OBJS += classifierddi.o classifier.o filters.o trie.o table.o \
512     ba_table.o

514 DSCPMK_OBJS += dscpmk.o dscpmkddi.o

516 DLCOSMK_OBJS += dlcosmk.o dlcosmkddi.o

518 FLOWACCT_OBJS += flowacctddi.o flowacct.o

520 TOKENMT_OBJS += tokenmt.o tokenmtddi.o

522 TSWTCL_OBJS += tswtcl.o tswtclddi.o

```

```

524 ARP_OBJS += arpddi.o
526 ICMP_OBJS += icmpddi.o
528 ICMP6_OBJS += icmp6ddi.o
530 RTS_OBJS += rtsddi.o

532 IP_ICMP_OBJS = icmp.o icmp_opt_data.o
533 IP_RTS_OBJS = rts.o rts_opt_data.o
534 IP_TCP_OBJS = tcp.o tcp_fusion.o tcp_opt_data.o tcp_sack.o tcp_stats.o \
535 tcp_misc.o tcp_timers.o tcp_time_wait.o tcp_tpi.o tcp_output.o \
536 tcp_input.o tcp_socket.o tcp_bind.o tcp_cluster.o tcp_tunables.o
537 IP_UDP_OBJS = udp.o udp_opt_data.o udp_tunables.o udp_stats.o
538 IP_SCTP_OBJS = sctp.o sctp_opt_data.o sctp_output.o \
539 sctp_init.o sctp_input.o sctp_cookie.o \
540 sctp_conn.o sctp_error.o sctp_snmp.o \
541 sctp_tunables.o sctp_shutdown.o sctp_common.o \
542 sctp_timer.o sctp_heartbeat.o sctp_hash.o \
543 sctp_bind.o sctp_notify.o sctp_asconf.o \
544 sctp_addr.o tn_ipopt.o tnet.o ip_netinfo.o \
545 sctp_misc.o
546 IP_ILB_OBJS = ilb.o ilb_nat.o ilb_conn.o ilb_alg_hash.o ilb_alg_rr.o

548 IP_OBJS += igmp.o ipmp.o ip.o ip6.o ip6_asp.o ip6_if.o ip6_ire.o \
549 ip6_rts.o ip_if.o ip_ire.o ip_listutils.o ip_mrout.o \
550 ip_multi.o ip2mac.o ip_ndp.o ip_rts.o ip_srcid.o \
551 ipddi.o ipdrop.o mi.o nd.o tunables.o optcom.o snmpcom.o \
552 ipsec_loader.o spd.o ipclassifier.o inet_common.o ip_queue.o \
553 queue.o ip_sadb.o ip_ftable.o proto_set.o radix.o ip_dummy.o \
554 ip_helper_stream.o ip_tunables.o \
555 ip_output.o ip_input.o ip6_input.o ip6_output.o ip_arp.o \
556 conn_opt.o ip_attr.o ip_dce.o \
557 $(IP_ICMP_OBJS) \
558 $(IP_RTS_OBJS) \
559 $(IP_TCP_OBJS) \
560 $(IP_UDP_OBJS) \
561 $(IP_SCTP_OBJS) \
562 $(IP_ILB_OBJS)

564 IP6_OBJS += ip6ddi.o

566 HOOK_OBJS += hook.o

568 NETI_OBJS += neti_impl.o neti_mod.o neti_stack.o

570 KEYSOCK_OBJS += keysockddi.o keysock.o keysock_opt_data.o

572 IPNET_OBJS += ipnet.o ipnet_bpf.o

574 SPDSOCK_OBJS += spdsockddi.o spdsock.o spdsock_opt_data.o

576 IPSECESP_OBJS += ipsecespddi.o ipsecesp.o

578 IPSECAH_OBJS += ipsecahddi.o ipsecah.o sadb.o

580 SPPP_OBJS += sPPP.o sPPP_dlpi.o sPPP_mod.o s_common.o

582 SPPTUN_OBJS += spptun.o spptun_mod.o

584 SPPASYN_OBJS += spppasyn.o spppasyn_mod.o

586 SPPPCOMP_OBJS += spppcomp.o spppcomp_mod.o deflate.o BSD-comp.o vjcompress.o \
587 zlib.o

589 TCP_OBJS += tcpddi.o

```

```

591 TCP6_OBJS += tcp6ddi.o

593 NCA_OBJS += ncaddi.o

595 SDP SOCK_MOD_OBJS += sockmod_sdp.o socksdp.o socksdpsubr.o

597 SCTP SOCK_MOD_OBJS += sockmod_sctp.o socksctp.o socksctpsubr.o

599 PFP SOCK_MOD_OBJS += sockmod_pfp.o

601 RDS SOCK_MOD_OBJS += sockmod_rds.o

603 RDS_OBJS += rdsddi.o rdssubr.o rds_opt.o rds_ioctl.o

605 RDSIB_OBJS += rdsib.o rdsib_ib.o rdsib_cm.o rdsib_ep.o rdsib_buf.o \
606 rdsib_debug.o rdsib_sc.o

608 RDSV3_OBJS += af_rds.o rdsV3_ddi.o bind.o loop.o threads.o connection.o \
609 transport.o cong.o sysctl.o message.o rds_recv.o send.o \
610 stats.o info.o page.o rdma_transport.o ib_ring.o ib_rdma.o \
611 ib_recv.o ib.o ib_send.o ib_sysctl.o ib_stats.o ib_cm.o \
612 rdsV3_sc.o rdsV3_debug.o rdsV3_impl.o rdma.o rdsV3_af_thr.o

614 ISER_OBJS += iser.o iser_cm.o iser_cq.o iser_ib.o iser_idm.o \
615 iser_resource.o iser_xfer.o

617 UDP_OBJS += udpddi.o

619 UDP6_OBJS += udp6ddi.o

621 SY_OBJS += gentyty.o

623 TCO_OBJS += ticots.o

625 TCOO_OBJS += ticotsord.o

627 TCL_OBJS += ticlts.o

629 TL_OBJS += tl.o

631 DUMP_OBJS += dump.o

633 BPF_OBJS += bpf.o bpf_filter.o bpf_mod.o bpf_dlt.o bpf_mac.o

635 CLONE_OBJS += clone.o

637 CN_OBJS += cons.o

639 DLD_OBJS += dld_drv.o dld_proto.o dld_str.o dld_flow.o

641 DLS_OBJS += dls.o dls_link.o dls_mod.o dls_stat.o dls_mgmt.o

643 GLD_OBJS += gld.o gldutil.o

645 MAC_OBJS += mac.o mac_bcast.o mac_client.o mac_datapath_setup.o mac_flow.o
646 mac_hio.o mac_mod.o mac_ndd.o mac_provider.o mac_sched.o \
647 mac_protect.o mac_soft_ring.o mac_stat.o mac_util.o

649 MAC_6TO4_OBJS += mac_6to4.o

651 MAC_ETHER_OBJS += mac_ether.o

653 MAC_IPV4_OBJS += mac_ipv4.o

655 MAC_IPV6_OBJS += mac_ipv6.o

```



```

657 MAC_WIFI_OBJS +=      mac_wifi.o
659 MAC_IB_OBJS +=       mac_ib.o
661 IPTUN_OBJS +=       iptun_dev.o iptun_ctl.o iptun.o
663 AGGR_OBJS +=        aggr_dev.o aggr_ctl.o aggr_grp.o aggr_port.o \
664                      aggr_send.o aggr_recv.o aggr_lacp.o
666 SOFTMAC_OBJS +=     softmac_main.o softmac_ctl.o softmac_capab.o \
667                      softmac_dev.o softmac_stat.o softmac_pkt.o softmac_fp.o
669 NET80211_OBJS +=    net80211.o net80211_proto.o net80211_input.o \
670                      net80211_output.o net80211_node.o net80211_crypto.o \
671                      net80211_crypto_none.o net80211_crypto_wep.o net80211_ioctl.o \
672                      net80211_crypto_tkip.o net80211_crypto_ccmp.o \
673                      net80211_ht.o
675 VNIC_OBJS +=       vnic_ctl.o vnic_dev.o
677 SIMNET_OBJS +=     simnet.o
679 IB_OBJS +=         ibnex.o ibnex_ioctl.o ibnex_hca.o
681 IBCM_OBJS +=       ibcm_impl.o ibcm_sm.o ibcm_ti.o ibcm_utils.o ibcm_path.o \
682                      ibcm_arp.o ibcm_arp_link.o
684 IBDM_OBJS +=       ibdm.o
686 IBDMA_OBJS +=      ibdma.o
688 IBMF_OBJS +=       ibmf.o ibmf_impl.o ibmf_dr.o ibmf_wqe.o ibmf_ud_dest.o ibmf_mod.o
689                      ibmf_send.o ibmf_recv.o ibmf_handlers.o ibmf_trans.o \
690                      ibmf_timers.o ibmf_msg.o ibmf_utils.o ibmf_rmpp.o \
691                      ibmf_saa.o ibmf_saa_impl.o ibmf_saa_utils.o ibmf_saa_events.o
693 IBTL_OBJS +=       ibtl_impl.o ibtl_util.o ibtl_mem.o ibtl_handlers.o ibtl_qp.o \
694                      ibtl_cg.o ibtl_wr.o ibtl_hca.o ibtl_chan.o ibtl_cm.o \
695                      ibtl_mcg.o ibtl_ibnex.o ibtl_srqp.o ibtl_part.o
697 TAVOR_OBJS +=      tavor.o tavor_agents.o tavor_cfg.o tavor_ci.o tavor_cmd.o \
698                      tavor_cq.o tavor_event.o tavor_ioctl.o tavor_misc.o \
699                      tavor_mr.o tavor_qp.o tavor_qpmod.o tavor_rsrc.o \
700                      tavor_srqp.o tavor_stats.o tavor_umap.o tavor_wr.o
702 HERMON_OBJS +=     hermon.o hermon_agents.o hermon_cfg.o hermon_ci.o hermon_cmd.o \
703                      hermon_cq.o hermon_event.o hermon_ioctl.o hermon_misc.o \
704                      hermon_mr.o hermon_qp.o hermon_qpmod.o hermon_rsrc.o \
705                      hermon_srqp.o hermon_stats.o hermon_umap.o hermon_wr.o \
706                      hermon_fcoib.o hermon_fm.o
708 DAPLT_OBJS +=      daplt.o
710 SOL_OFS_OBJS +=    sol_cma.o sol_ib_cma.o sol_uobj.o \
711                      sol_ofs_debug_util.o sol_ofs_gen_util.o \
712                      sol_kverbs.o
714 SOL_UCMA_OBJS +=   sol_ucma.o
716 SOL_UVERBS_OBJS += sol_uverbs.o sol_uverbs_comp.o sol_uverbs_event.o \
717                      sol_uverbs_hca.o sol_uverbs_qp.o
719 SOL_UMAD_OBJS +=   sol_umad.o
721 KSTAT_OBJS +=     kstat.o

```

```

723 KSYMS_OBJS +=      ksyms.o
725 INSTANCE_OBJS +=   inst_sync.o
727 IWSCN_OBJS +=      iwscons.o
729 LOFI_OBJS +=       lofi.o LzmaDec.o
731 FSSNAP_OBJS +=     fssnap.o
733 FSSNAPIF_OBJS +=   fssnap_if.o
735 MM_OBJS +=         mem.o
737 PHYSMEM_OBJS +=    physmem.o
739 OPTIONS_OBJS +=    options.o
741 WINLOCK_OBJS +=    winlockio.o
743 PM_OBJS +=         pm.o
744 SRN_OBJS +=        srn.o
746 PSEUDO_OBJS +=     pseudonex.o
748 RAMDISK_OBJS +=    ramdisk.o
750 LLC1_OBJS +=       llcl.o
752 USBKBM_OBJS +=     usbkbm.o
754 USBWCM_OBJS +=     usbwcm.o
756 BOFI_OBJS +=       bofi.o
758 HID_OBJS +=        hid.o
760 HWA_RC_OBJS +=     hwarc.o
762 USBSKEL_OBJS +=    usbskel.o
764 USBVC_OBJS +=      usbvc.o usbvc_v412.o
766 HIDPARSER_OBJS += hidparser.o
768 USB_AC_OBJS +=     usb_ac.o
770 USB_AS_OBJS +=     usb_as.o
772 USB_AH_OBJS +=     usb_ah.o
774 USBMS_OBJS +=      usbms.o
776 USBPRN_OBJS +=     usbprn.o
778 UGEN_OBJS +=       ugen.o
780 USBSER_OBJS +=     usbser.o usbser_rseq.o
782 USBSACM_OBJS +=    usb sacram.o
784 USBSER_KEYSPAN_OBJS += usbser_keyspan.o keyspan_dsd.o keyspan_pipe.o
786 USBS49_FW_OBJS +=  keyspan_49fw.o

```

## new/usr/src/uts/common/Makefile.files

13

```

788 USBSPRL_OBJS += usbser_pl2303.o pl2303_dsd.o
790 WUSB_CA_OBJS += wusb_ca.o
792 USBFTDI_OBJS += usbser_uftdi.o uftdi_dsd.o
794 USBECM_OBJS += usbecm.o
796 WC_OBJS += wscons.o vcons.o
798 VCONS_CONF_OBJS += vcons_conf.o
800 SCSI_OBJS +=      scsi_capabilities.o scsi_confsubr.o scsi_control.o \
801                 scsi_data.o scsi_fm.o scsi_hba.o scsi_reset_notify.o \
802                 scsi_resource.o scsi_subr.o scsi_transport.o scsi_watch.o \
803                 smp_transport.o
805 SCSI_VHCI_OBJS +=      scsi_vhci.o mpapi_impl.o scsi_vhci_tpgs.o
807 SCSI_VHCI_F_SYM_OBJS +=      sym.o
809 SCSI_VHCI_F_TPGS_OBJS +=      tpgs.o
811 SCSI_VHCI_F_ASYM_SUN_OBJS +=  asym_sun.o
813 SCSI_VHCI_F_SYM_HDS_OBJS +=  sym_hds.o
815 SCSI_VHCI_F_TAPE_OBJS +=      tape.o
817 SCSI_VHCI_F_TPGS_TAPE_OBJS += tpgs_tape.o
819 SGEN_OBJS +=      sgen.o
821 SMP_OBJS +=      smp.o
823 SATA_OBJS +=      sata.o
825 USBA_OBJS +=      hcidi.o  usba.o  usbai.o  hubdi.o  parser.o  genconsole.o \
826                 usbai_pipe_mgmt.o usbai_req.o usbai_util.o usbai_register.o \
827                 usba_devdb.o usba10_calls.o usba_uugen.o whcdi.o wa.o
828 USBA_WITHOUT_WUSB_OBJS +=      hcidi.o  usba.o  usbai.o  hubdi.o  parser.o  gencons
829                 usbai_pipe_mgmt.o usbai_req.o usbai_util.o usbai_register.o \
830                 usba_devdb.o usba10_calls.o usba_uugen.o
832 USBA10_OBJS +=      usba10.o
834 RSM_OBJS +=      rsm.o  rsmka_pathmanager.o  rsmka_util.o
836 RSMOPS_OBJS +=      rsmops.o
838 S1394_OBJS +=      t1394.o t1394_errmsg.o s1394.o s1394_addr.o s1394_async.o \
839                 s1394_bus_reset.o s1394_cmp.o s1394_csr.o s1394_dev_disc.o \
840                 s1394_fa.o s1394_fcp.o \
841                 s1394_hotplug.o s1394_isoch.o s1394_misc.o h1394.o nx1394.o
843 HCI1394_OBJS +=      hcil1394.o hcil1394_async.o hcil1394_attach.o hcil1394_buf.o \
844                 hcil1394_csr.o hcil1394_detach.o hcil1394_extern.o \
845                 hcil1394_ioctl.o hcil1394_isoch.o hcil1394_isr.o \
846                 hcil1394_ixl_comp.o hcil1394_ixl_isr.o hcil1394_ixl_misc.o \
847                 hcil1394_ixl_update.o hcil1394_misc.o hcil1394_ohci.o \
848                 hcil1394_q.o hcil1394_s1394if.o hcil1394_tlabel.o \
849                 hcil1394_tlist.o hcil1394_vendor.o
851 AV1394_OBJS +=      avl1394.o avl1394_as.o avl1394_async.o avl1394_cfgrom.o \
852                 avl1394_cmp.o avl1394_fcp.o avl1394_isoch.o avl1394_isoch_chan.o \
853                 avl1394_isoch_recv.o avl1394_isoch_xmit.o avl1394_list.o \

```

## new/usr/src/uts/common/Makefile.files

14

```

854                 avl1394_queue.o
856 DCAM1394_OBJS +=      dcam.o dcam_frame.o dcam_param.o dcam_reg.o \
857                 dcam_ring_buff.o
859 SCSA1394_OBJS +=      hba.o  sbp2_driver.o  sbp2_bus.o
861 SBP2_OBJS +=      cfgrom.o  sbp2.o
863 PMODEM_OBJS +=      pmodem.o pmodem_cis.o cis.o cis_callout.o cis_handlers.o cis_para
865 DSW_OBJS +=      dsw.o dsw_dev.o ii_tree.o
867 NCALL_OBJS +=      ncall.o \
868                 ncall_stub.o
870 RDC_OBJS +=      rdc.o \
871                 rdc_dev.o \
872                 rdc_io.o \
873                 rdc_clnt.o \
874                 rdc_prot_xdr.o \
875                 rdc_svc.o \
876                 rdc_bitmap.o \
877                 rdc_health.o \
878                 rdc_subr.o \
879                 rdc_diskq.o
881 RDCSRV_OBJS +=      rdcsrv.o
883 RDCSTUB_OBJS +=      rdc_stub.o
885 SDBC_OBJS +=      sd_bcache.o \
886                 sd_bio.o \
887                 sd_conf.o \
888                 sd_ft.o \
889                 sd_hash.o \
890                 sd_io.o \
891                 sd_misc.o \
892                 sd_pcu.o \
893                 sd_tdaemon.o \
894                 sd_trace.o \
895                 sd_iob_impl0.o \
896                 sd_iob_impl1.o \
897                 sd_iob_impl2.o \
898                 sd_iob_impl3.o \
899                 sd_iob_impl4.o \
900                 sd_iob_impl5.o \
901                 sd_iob_impl6.o \
902                 sd_iob_impl7.o \
903                 safestore.o \
904                 safestore_ram.o
906 NSCTL_OBJS +=      nsctl.o \
907                 nsc_cache.o \
908                 nsc_disk.o \
909                 nsc_dev.o \
910                 nsc_freeze.o \
911                 nsc_gen.o \
912                 nsc_mem.o \
913                 nsc_ncallio.o \
914                 nsc_power.o \
915                 nsc_resv.o \
916                 nsc_rmspin.o \
917                 nsc_solaris.o \
918                 nsc_trap.o \
919                 nsc_list.o

```

```

920 UNISTAT_OBJS += spuni.o \
921                spcs_s_k.o

923 NSKERN_OBJS += nsc_ddi.o \
924                nsc_proc.o \
925                nsc_raw.o \
926                nsc_thread.o \
927                nskernd.o

929 SV_OBJS += sv.o

931 PMCS_OBJS += pmcs_attach.o pmcs_ds.o pmcs_intr.o pmcs_nvram.o pmcs_sata.o \
932             pmcs_scsa.o pmcs_smhba.o pmcs_subr.o pmcs_fwlog.o

934 PMCS8001FW_C_OBJS += pmcs_fw_hdr.o
935 PMCS8001FW_OBJS += $(PMCS8001FW_C_OBJS) SPCBoot.o ila.o firmware.o

937 FSD_OBJS += fsd.o

939 #
940 #      Build up defines and paths.

942 ST_OBJS += st.o st_conf.o

944 EMLXS_OBJS += emlxs_clock.o emlxs_dfc.o emlxs_dhchap.o emlxs_diag.o \
945              emlxs_download.o emlxs_dump.o emlxs_els.o emlxs_event.o \
946              emlxs_fcf.o emlxs_fcp.o emlxs_fct.o emlxs_hba.o emlxs_ip.o \
947              emlxs_mbox.o emlxs_mem.o emlxs_msg.o emlxs_node.o \
948              emlxs_pkt.o emlxs_sli3.o emlxs_sli4.o emlxs_solaris.o \
949              emlxs_thread.o

951 EMLXS_FW_OBJS += emlxs_fw.o

953 OCE_OBJS += oce_buf.o oce_fm.o oce_gld.o oce_hw.o oce_intr.o oce_main.o \
954            oce_mbx.o oce_mq.o oce_queue.o oce_rx.o oce_stat.o oce_tx.o \
955            oce_utils.o

957 FCT_OBJS += discovery.o fct.o

959 QLT_OBJS += 2400.o 2500.o 8100.o qlt.o qlt_dma.o

961 SRPT_OBJS += srpt_mod.o srpt_ch.o srpt_cm.o srpt_ioc.o srpt_stp.o

963 FCOE_OBJS += fcoe.o fcoe_eth.o fcoe_fc.o

965 FCOET_OBJS += fcoet.o fcoet_eth.o fcoet_fc.o

967 FCOEI_OBJS += fcoei.o fcoei_eth.o fcoei_lv.o

969 ISCSIT_SHARED_OBJS += \
970                    iscsit_common.o

972 ISCSIT_OBJS += $(ISCSIT_SHARED_OBJS) \
973              iscsit.o iscsit_tgt.o iscsit_sess.o iscsit_login.o \
974              iscsit_text.o iscsit_isns.o iscsit_radiusauth.o \
975              iscsit_radiuspacket.o iscsit_auth.o iscsit_authclient.o

977 PPPT_OBJS += alua_ic_if.o pppt.o pppt_msg.o pppt_tgt.o

979 STMF_OBJS += lun_map.o stmf.o

981 STMF_SBD_OBJS += sbd.o sbd_scsi.o sbd_pgr.o sbd_zvol.o

983 SYMSG_OBJS += sysmsg.o

985 SES_OBJS += ses.o ses_sen.o ses_safte.o ses_ses.o

```

```

987 TNF_OBJS += tnf_buf.o tnf_trace.o tnf_writer.o trace_init.o \
988            trace_funcs.o tnf_probe.o tnf.o

990 LOGINDMUX_OBJS += logindmux.o

992 DEVINFO_OBJS += devinfo.o

994 DEVPOLL_OBJS += devpoll.o

996 DEVPOOL_OBJS += devpool.o

998 I8042_OBJS += i8042.o

1000 KB8042_OBJS += \
1001              at_keyprocess.o \
1002              kb8042.o \
1003              kb8042_keytables.o

1005 MOUSE8042_OBJS += mouse8042.o

1007 FDC_OBJS += fdc.o

1009 ASY_OBJS += asy.o

1011 ECPP_OBJS += ecpp.o

1013 VUIDM3P_OBJS += vuidmice.o vuidm3p.o

1015 VUIDM4P_OBJS += vuidmice.o vuidm4p.o

1017 VUIDM5P_OBJS += vuidmice.o vuidm5p.o

1019 VUIDPS2_OBJS += vuidmice.o vuidps2.o

1021 HPCSV_C_OBJS += hpcsvc.o

1023 PCIE_MISC_OBJS += pcie.o pcie_fault.o pcie_hp.o pciehp.o pcishpc.o pcie_pwr.o p

1025 PCIHPNEXUS_OBJS += pcihp.o

1027 OPENEPR_OBJS += openprom.o

1029 RANDOM_OBJS += random.o

1031 PSHOT_OBJS += pshot.o

1033 GEN_DRV_OBJS += gen_drv.o

1035 TCLIENT_OBJS += tclient.o

1037 TPHCI_OBJS += tphci.o

1039 TVHCI_OBJS += tvhci.o

1041 EMUL64_OBJS += emul64.o emul64_bsd.o

1043 FCP_OBJS += fcp.o

1045 FCIP_OBJS += fcip.o

1047 FCSM_OBJS += fcsm.o

1049 FCTL_OBJS += fctl.o

1051 FP_OBJS += fp.o

```

```

1053 QLC_OBJS += ql_api.o ql_debug.o ql_hba_fru.o ql_init.o ql_iocb.o ql_ioctl.o \
1054     ql_isr.o ql_mbx.o ql_nx.o ql_xioctl.o ql_fw_table.o

1056 QLC_FW_2200_OBJS += ql_fw_2200.o

1058 QLC_FW_2300_OBJS += ql_fw_2300.o

1060 QLC_FW_2400_OBJS += ql_fw_2400.o

1062 QLC_FW_2500_OBJS += ql_fw_2500.o

1064 QLC_FW_6322_OBJS += ql_fw_6322.o

1066 QLC_FW_8100_OBJS += ql_fw_8100.o

1068 QLGE_OBJS += qlge.o qlge_dbg.o qlge_flash.o qlge_fm.o qlge_gld.o qlge_mpi.o

1070 ZCONS_OBJS += zcons.o

1072 NV_SATA_OBJS += nv_sata.o

1074 SI3124_OBJS += si3124.o

1076 AHCI_OBJS += ahci.o

1078 PCIIDE_OBJS += pci-ide.o

1080 PCEPP_OBJS += pcepp.o

1082 CPC_OBJS += cpc.o

1084 CPUID_OBJS += cpuid_drv.o

1086 SYSEVENT_OBJS += sysevent.o

1088 BL_OBJS += bl.o

1090 DRM_OBJS += drm_sunmod.o drm_kstat.o drm_agpsupport.o \
1091     drm_auth.o drm_bufs.o drm_context.o drm_dma.o \
1092     drm_drawable.o drm_drv.o drm_fops.o drm_ioctl.o drm_irq.o \
1093     drm_lock.o drm_memory.o drm_msg.o drm_pci.o drm_scatter.o \
1094     drm_cache.o drm_gem.o drm_mm.o ati_pigart.o

1096 FM_OBJS += devfm.o devfm_machdep.o

1098 RTLS_OBJS += rtls.o

1100 #
1101 #             exec modules
1102 #
1103 AOUTEXEC_OBJS +=aout.o

1105 ELFEXEC_OBJS += elf.o elf_notes.o old_notes.o

1107 INTPEXEC_OBJS +=intp.o

1109 SHBINEXEC_OBJS +=shbin.o

1111 JAVAEXEC_OBJS +=java.o

1113 #
1114 #             file system modules
1115 #
1116 AUTOFS_OBJS += auto_vfsops.o auto_vnops.o auto_subr.o auto_xdr.o auto_sys.o

```

```

1118 CACHEFS_OBJS += cachefs_cnode.o           cachefs_cod.o \
1119     cachefs_dir.o           cachefs_dlog.o  cachefs_filegrp.o \
1120     cachefs_fscache.o      cachefs_ioctl.o cachefs_log.o \
1121     cachefs_module.o \
1122     cachefs_noopc.o        cachefs_resource.o \
1123     cachefs_strict.o \
1124     cachefs_subr.o         cachefs_vfsops.o \
1125     cachefs_vnops.o

1127 DCFS_OBJS += dc_vnops.o

1129 DEVFS_OBJS += devfs_subr.o  devfs_vfsops.o devfs_vnops.o

1131 DEV_OBJS  += sdev_subr.o     sdev_vfsops.o sdev_vnops.o \
1132     sdev_ptsops.o sdev_zvolops.o sdev_comm.o \
1133     sdev_profile.o sdev_ncache.o sdev_netops.o \
1134     sdev_ipnetops.o \
1135     sdev_vttops.o

1137 CTFS_OBJS += ctfs_all.o ctfs_cdir.o ctfs_ctl.o ctfs_event.o \
1138     ctfs_latest.o ctfs_root.o ctfs_sym.o ctfs_tdir.o ctfs_tmpl.o

1140 OBJFS_OBJS += objfs_vfs.o  objfs_root.o  objfs_common.o \
1141     objfs_odir.o  objfs_data.o

1143 FDFS_OBJS += fdops.o

1145 FIFO_OBJS += fifosubr.o  fifovnops.o

1147 PIPE_OBJS += pipe.o

1149 HSFS_OBJS += hsfs_node.o  hsfs_subr.o  hsfs_vfsops.o  hsfs_vnops.o \
1150     hsfs_susp.o  hsfs_rrip.o  hsfs_susp_subr.o

1152 LOFS_OBJS += lofs_subr.o  lofs_vfsops.o  lofs_vnops.o

1154 NAMEFS_OBJS += namevfs.o  namevno.o

1156 NFS_OBJS += nfs_client.o  nfs_common.o  nfs_dump.o \
1157     nfs_subr.o  nfs_vfsops.o  nfs_vnops.o \
1158     nfs_xdr.o  nfs_sys.o  nfs_strerror.o \
1159     nfs3_vfsops.o  nfs3_vnops.o  nfs3_xdr.o \
1160     nfs_acl_vnops.o  nfs_acl_xdr.o  nfs4_vfsops.o \
1161     nfs4_vnops.o  nfs4_xdr.o  nfs4_idmap.o \
1162     nfs4_shadow.o  nfs4_subr.o \
1163     nfs4_attr.o  nfs4_rnode.o  nfs4_client.o \
1164     nfs4_acache.o  nfs4_common.o  nfs4_client_state.o \
1165     nfs4_callback.o  nfs4_recovery.o  nfs4_client_secinfo.o \
1166     nfs4_client_debug.o  nfs_stats.o \
1167     nfs4_acl.o  nfs4_stub_vnops.o  nfs_cmd.o

1169 NFSSRV_OBJS += nfs_server.o  nfs_srv.o  nfs3_srv.o \
1170     nfs_acl_srv.o  nfs_auth.o  nfs_auth_xdr.o \
1171     nfs_export.o  nfs_log.o  nfs_log_xdr.o \
1172     nfs4_srv.o  nfs4_state.o  nfs4_srv_attr.o \
1173     nfs4_srv_ns.o  nfs4_db.o  nfs4_srv_deleg.o \
1174     nfs4_deleg_ops.o  nfs4_srv_readdir.o  nfs4_dispatch.o

1176 SMBSRV_SHARED_OBJS += \
1177     smb_inet.o \
1178     smb_match.o \
1179     smb_msgbuf.o \
1180     smb_oem.o \
1181     smb_string.o \
1182     smb_utf8.o \
1183     smb_door_legacy.o \

```

## new/usr/src/uts/common/Makefile.files

```

1184         smb_xdr.o \
1185         smb_token.o \
1186         smb_token_xdr.o \
1187         smb_sid.o \
1188         smb_native.o \
1189         smb_netbios_util.o

1191 SMBSRV_OBJS += $(SMBSRV_SHARED_OBJS) \
1192         smb_acl.o \
1193         smb_alloc.o \
1194         smb_close.o \
1195         smb_common_open.o \
1196         smb_common_transact.o \
1197         smb_create.o \
1198         smb_delete.o \
1199         smb_directory.o \
1200         smb_dispatch.o \
1201         smb_echo.o \
1202         smb_fem.o \
1203         smb_find.o \
1204         smb_flush.o \
1205         smb_fsinfo.o \
1206         smb_fsops.o \
1207         smb_init.o \
1208         smb_kdoor.o \
1209         smb_kshare.o \
1210         smb_kutil.o \
1211         smb_lock.o \
1212         smb_lock_byte_range.o \
1213         smb_locking_andx.o \
1214         smb_logoff_andx.o \
1215         smb_mangle_name.o \
1216         smb_mbuf_marshall.o \
1217         smb_mbuf_util.o \
1218         smb_negotiate.o \
1219         smb_net.o \
1220         smb_node.o \
1221         smb_nt_cancel.o \
1222         smb_nt_create_andx.o \
1223         smb_nt_transact_create.o \
1224         smb_nt_transact_ioctl.o \
1225         smb_nt_transact_notify_change.o \
1226         smb_nt_transact_quota.o \
1227         smb_nt_transact_security.o \
1228         smb_odir.o \
1229         smb_ofile.o \
1230         smb_open_andx.o \
1231         smb_opipe.o \
1232         smb_oplock.o \
1233         smb_pathname.o \
1234         smb_print.o \
1235         smb_process_exit.o \
1236         smb_query_fileinfo.o \
1237         smb_read.o \
1238         smb_rename.o \
1239         smb_sd.o \
1240         smb_seek.o \
1241         smb_server.o \
1242         smb_session.o \
1243         smb_session_setup_andx.o \
1244         smb_set_fileinfo.o \
1245         smb_signing.o \
1246         smb_tree.o \
1247         smb_trans2_create_directory.o \
1248         smb_trans2_dfs.o \
1249         smb_trans2_find.o

```

19

## new/usr/src/uts/common/Makefile.files

```

1250         smb_tree_connect.o \
1251         smb_unlock_byte_range.o \
1252         smb_user.o \
1253         smb_vfs.o \
1254         smb_vops.o \
1255         smb_vss.o \
1256         smb_write.o \
1257         smb_write_raw.o

1259 PCFS_OBJS += pc_alloc.o pc_dir.o pc_node.o pc_subr.o \
1260         pc_vfsops.o pc_vnops.o

1262 PROC_OBJS += prcontrol.o prioctl.o prsubr.o prusr.o \
1263         prvnops.o

1265 MNTFS_OBJS += mntvfsops.o mntvnops.o

1267 SHAREFS_OBJS += sharetab.o sharefs_vfsops.o sharefs_vnops.o

1269 SPEC_OBJS += specsubr.o specvops.o specvnops.o

1271 SOCK_OBJS += socksubr.o sockvops.o sockparams.o \
1272         socksyscalls.o socktpi.o sockstr.o \
1273         sockcommon_vnops.o sockcommon_subr.o \
1274         sockcommon_sops.o sockcommon.o \
1275         sock_notsupp.o socknotify.o \
1276         nl7c.o nl7curi.o nl7chttp.o nl7clogd.o \
1277         nl7cnca.o sodirect.o sockfilter.o

1279 TMPFS_OBJS += tmp_dir.o tmp_subr.o tmp_tnode.o tmp_vfsops.o \
1280         tmp_vnops.o

1282 UDFS_OBJS += udf_alloc.o udf_bmap.o udf_dir.o \
1283         udf_inode.o udf_subr.o udf_vfsops.o \
1284         udf_vnops.o

1286 UFS_OBJS += ufs_alloc.o ufs_bmap.o ufs_dir.o ufs_xattr.o \
1287         ufs_inode.o ufs_subr.o ufs_tables.o ufs_vfsops.o \
1288         ufs_vnops.o quota.o quotacalls.o quota_ufs.o \
1289         ufs_filio.o ufs_lockfs.o ufs_thread.o ufs_trans.o \
1290         ufs_acl.o ufs_panic.o ufs_directio.o ufs_log.o \
1291         ufs_extvnops.o ufs_snap.o lufs.o lufs_thread.o \
1292         lufs_log.o lufs_map.o lufs_top.o lufs_debug.o \
1293         vscan_drv.o vscan_svc.o vscan_door.o

1295 NSMB_OBJS += smb_conn.o smb_dev.o smb_iod.o smb_pass.o \
1296         smb_rq.o smb_sign.o smb_smb.o smb_subrs.o \
1297         smb_time.o smb_tran.o smb_trantcp.o smb_usr.o \
1298         subr_mchain.o

1300 SMBFS_COMMON_OBJS += smbfs_ntacl.o
1301 SMBFS_OBJS += smbfs_vfsops.o smbfs_vnops.o smbfs_node.o \
1302         smbfs_acl.o smbfs_client.o smbfs_smb.o \
1303         smbfs_subr.o smbfs_subr2.o \
1304         smbfs_rwlock.o smbfs_xattr.o \
1305         $(SMBFS_COMMON_OBJS)

1308 #
1309 #           LVM modules
1310 #
1311 MD_OBJS += md.o md_error.o md_ioctl.o md_mddb.o md_names.o \
1312         md_med.o md_rename.o md_subr.o

1314 MD_COMMON_OBJS = md_convert.o md_crc.o md_revchk.o

```

20

## new/usr/src/uts/common/Makefile.files

21

```

1316 MD_DERIVED_OBJS = metamed_xdr.o meta_basic_xdr.o
1318 SOFTPART_OBJS += sp.o sp_ioctl.o
1320 STRIPE_OBJS += stripe.o stripe_ioctl.o
1322 HOTSPARES_OBJS += hotspares.o
1324 RAID_OBJS += raid.o raid_ioctl.o raid_replay.o raid_resync.o raid_hotspare.o
1326 MIRROR_OBJS += mirror.o mirror_ioctl.o mirror_resync.o
1328 NOTIFY_OBJS += md_notify.o
1330 TRANS_OBJS += mdtrans.o trans_ioctl.o trans_log.o
1332 ZFS_COMMON_OBJS += \
1333     arc.o \
1334     bplist.o \
1335     bpobj.o \
1336     bptree.o \
1337     dbuf.o \
1338     ddt.o \
1339     ddt_zap.o \
1340     dmuf.o \
1341     dmuf_diff.o \
1342     dmuf_send.o \
1343     dmuf_object.o \
1344     dmuf_objset.o \
1345     dmuf_traverse.o \
1346     dmuf_tx.o \
1347     dnode.o \
1348     dnode_sync.o \
1349     dsl_dir.o \
1350     dsl_dataset.o \
1351     dsl_deadlist.o \
1352     dsl_destroy.o \
1353     dsl_pool.o \
1354     dsl_synctask.o \
1355     dsl_userhold.o \
1356     dmuf_zfetch.o \
1357     dsl_deleg.o \
1358     dsl_prop.o \
1359     dsl_scan.o \
1360     zfeature.o \
1361     gzip.o \
1362     lz4.o \
1363     lzjb.o \
1364     metaslab.o \
1365     refcount.o \
1366     rrwlock.o \
1367     sa.o \
1368     sha256.o \
1369     spa.o \
1370     spa_config.o \
1371     spa_errlog.o \
1372     spa_history.o \
1373     spa_misc.o \
1374     space_map.o \
1375     txg.o \
1376     uberblock.o \
1377     unique.o \
1378     vdev.o \
1379     vdev_cache.o \
1380     vdev_file.o \
1381     vdev_label.o \

```

## new/usr/src/uts/common/Makefile.files

22

```

1382     vdev_mirror.o \
1383     vdev_missing.o \
1384     vdev_queue.o \
1385     vdev_raidz.o \
1386     vdev_root.o \
1387     zap.o \
1388     zap_leaf.o \
1389     zap_micro.o \
1390     zfs_byteswap.o \
1391     zfs_debug.o \
1392     zfs_fm.o \
1393     zfs_fuid.o \
1394     zfs_sa.o \
1395     zfs_znode.o \
1396     zil.o \
1397     zio.o \
1398     zio_checksum.o \
1399     zio_compress.o \
1400     zio_inject.o \
1401     zle.o \
1402     zlock.o
1404 ZFS_SHARED_OBJS += \
1405     zfeature_common.o \
1406     zfs_comutil.o \
1407     zfs_deleg.o \
1408     zfs_fletcher.o \
1409     zfs_namecheck.o \
1410     zfs_prop.o \
1411     zpool_prop.o \
1412     zprop_common.o
1414 ZFS_OBJS += \
1415     $(ZFS_COMMON_OBJS) \
1416     $(ZFS_SHARED_OBJS) \
1417     vdev_disk.o \
1418     zfs_acl.o \
1419     zfs_ctldir.o \
1420     zfs_dir.o \
1421     zfs_ioctl.o \
1422     zfs_log.o \
1423     zfs_onexit.o \
1424     zfs_replay.o \
1425     zfs_rlock.o \
1426     zfs_vfsops.o \
1427     zfs_vnops.o \
1428     zvol.o
1430 ZUT_OBJS += \
1431     zut.o
1433 # \
1434 #     streams modules
1435 #
1436 BUFMOD_OBJS += bufmod.o
1438 CONNLD_OBJS += connld.o
1440 DEDUMP_OBJS += dedump.o
1442 DRCOMPAT_OBJS += drcompat.o
1444 LDLINUX_OBJS += ldlinux.o
1446 LDTERM_OBJS += ldterm.o uwidth.o

```

## new/usr/src/uts/common/Makefile.files

23

```

1448 PKCT_OBJS +=      pckt.o
1450 PFMOD_OBJS +=      pfmod.o
1452 PTEM_OBJS +=      ptem.o
1454 REDIRMOD_OBJS +=  strredirm.o
1456 TIMOD_OBJS +=      timod.o
1458 TIRDWR_OBJS +=     tirdwr.o
1460 TTCOMPAT_OBJS +=  ttcompat.o
1462 LOG_OBJS +=        log.o
1464 PIPEMOD_OBJS +=    pipemod.o
1466 RPCMOD_OBJS +=      rpcmod.o      clnt_cots.o      clnt_clts.o \
1467      clnt_gen.o      clnt_perr.o      mt_rpcinit.o      rpc_calmsg.o \
1468      rpc_prot.o      rpc_sztypes.o   rpc_subr.o      rpcb_prot.o \
1469      svc.o           svc_clts.o      svc_gen.o      svc_cots.o \
1470      rpcsys.o      xdr_sizeof.o   clnt_rdma.o     svc_rdma.o \
1471      xdr_rdma.o     rdma_subr.o     xdrdma_sizeof.o
1473 TLIMOD_OBJS +=      tlimod.o      t_kalloc.o      t_kbind.o      t_kclose.o \
1474      t_kconnect.o    t_kfree.o      t_kgtstate.o    t_kopen.o \
1475      t_krcvdat.o     t_ksndudat.o   t_kspoll.o     t_kunbind.o \
1476      t_kutil.o
1478 RLMOD_OBJS +=      rlmmod.o
1480 TELMOD_OBJS +=     telmod.o
1482 CRYPTMOD_OBJS +=   cryptmod.o
1484 KB_OBJS +=         kbd.o          keytables.o
1486 #
1487 #             ID mapping module
1488 #
1489 IDMAP_OBJS +=      idmap_mod.o      idmap_kapi.o      idmap_xdr.o      idmap_cache.o
1491 #
1492 #             scheduling class modules
1493 #
1494 SDC_OBJS +=        sysdc.o
1496 RT_OBJS +=         rt.o
1497 RT_DPTBL_OBJS +=  rt_dptbl.o
1499 TS_OBJS +=         ts.o
1500 TS_DPTBL_OBJS +=   ts_dptbl.o
1502 IA_OBJS +=        ia.o
1504 FSS_OBJS +=        fss.o
1506 FX_OBJS +=         fx.o
1507 FX_DPTBL_OBJS +=   fx_dptbl.o
1509 #
1510 #             Inter-Process Communication (IPC) modules
1511 #
1512 IPC_OBJS +=        ipc.o

```

## new/usr/src/uts/common/Makefile.files

24

```

1514 IPCMSG_OBJS +=     msg.o
1516 IPCSEM_OBJS +=     sem.o
1518 IPCSHM_OBJS +=     shm.o
1520 #
1521 #             bignum module
1522 #
1523 COMMON_BIGNUM_OBJS += bignum_mod.o bignumimpl.o
1525 BIGNUM_OBJS +=     $(COMMON_BIGNUM_OBJS) $(BIGNUM_PSR_OBJS)
1527 #
1528 #             kernel cryptographic framework
1529 #
1530 KCF_OBJS +=         kcf.o kcf_callprov.o kcf_cbufcall.o kcf_cipher.o kcf_crypto.o \
1531      kcf_cryptoadm.o kcf_ctxops.o kcf_digest.o kcf_dual.o \
1532      kcf_keys.o kcf_mac.o kcf_mech_tabs.o kcf_miscapi.o \
1533      kcf_object.o kcf_policy.o kcf_prov_lib.o kcf_prov_tabs.o \
1534      kcf_sched.o kcf_session.o kcf_sign.o kcf_spi.o kcf_verify.o \
1535      kcf_random.o modes.o ecb.o cbc.o ctr.o ccm.o gcm.o \
1536      fips_random.o
1538 CRYPTOADM_OBJS +=  cryptoadm.o
1540 CRYPTO_OBJS +=     crypto.o
1542 DPROV_OBJS +=      dprov.o
1544 DCA_OBJS +=         dca.o dca_3des.o dca_debug.o dca_dsa.o dca_kstat.o dca_rng.o \
1545      dca_rsa.o
1547 AESPROV_OBJS +=    aes.o aes_impl.o aes_modes.o
1549 ARCFOURPROV_OBJS += arcfour.o arcfour_crypt.o
1551 BLOWFISHPROV_OBJS += blowfish.o blowfish_impl.o
1553 ECCPROV_OBJS +=     ecc.o ec.o ec2_163.o ec2_mont.o ecdecode.o ecl_mult.o \
1554      ecp_384.o ecp_jac.o ec2_193.o ecl.o ecp_192.o ecp_521.o \
1555      ecp_jm.o ec2_233.o ecl_curve.o ecp_224.o ecp_aff.o \
1556      ecp_mont.o ec2_aff.o ec_naf.o ecl_gf.o ecp_256.o mp_gf2m.o \
1557      mpi.o mplogic.o mpmontg.o mpprime.o oid.o \
1558      secitem.o ec2_test.o ecp_test.o
1560 RSAPROV_OBJS +=     rsa.o rsa_impl.o pkcs1.o
1562 SWRANDPROV_OBJS += swrand.o
1564 #
1565 #             kernel SSL
1566 #
1567 KSSL_OBJS +=        kssl.o ksslioc1.o
1569 KSSL_SOCKFIL_MOD_OBJS += ksslfilter.o ksslapi.o ksslrec.o
1571 #
1572 #             misc. modules
1573 #
1575 C2AUDIT_OBJS +=     adr.o audit.o audit_event.o audit_io.o \
1576      audit_path.o audit_start.o audit_syscalls.o audit_token.o \
1577      audit_mem.o
1579 PCIC_OBJS +=        pcic.o

```

```

1581 RPCSEC_OBJS += secmod.o      sec_clnt.o      sec_svc.o      sec_gen.o \
1582                auth_des.o      auth_kern.o      auth_none.o      auth_loopb.o \
1583                authdesprt.o      authdesubr.o      authu_prot.o \
1584                key_call.o      key_prot.o      svc_authu.o      svcauthdes.o

1586 RPCSEC_GSS_OBJS += rpcsec_gssmod.o rpcsec_gss.o rpcsec_gss_misc.o \
1587                rpcsec_gss_utils.o svc_rpcsec_gss.o

1589 CONSCONFIG_OBJS += consconfig.o

1591 CONSCONFIG_DACF_OBJS += consconfig_dacf.o consplat.o

1593 TEM_OBJS += tem.o tem_safe.o 6x10.o 7x14.o 12x22.o

1595 KBTRANS_OBJS +=                \
1596                kbtrans.o          \
1597                kbtrans_keytables.o \
1598                kbtrans_polled.o    \
1599                kbtrans_streams.o  \
1600                usb_keytables.o

1602 KGSSD_OBJS += gssd_clnt_stubs.o gssd_handle.o gssd_prot.o \
1603                gss_display_name.o gss_release_name.o gss_import_name.o \
1604                gss_release_buffer.o gss_release_oid_set.o gen_oids.o gssdmod.o

1606 KGSSD_DERIVED_OBJS = gssd_xdr.o

1608 KGSS_DUMMY_OBJS += dmech.o

1610 KSOCKET_OBJS += ksocket.o ksocket_mod.o

1612 CRYPTO= cksumtypes.o decrypt.o encrypt.o encrypt_length.o etypes.o \
1613          nfold.o verify_checksum.o prng.o block_size.o make_checksum.o \
1614          checksum_length.o hmac.o default_state.o mandatory_sumtype.o

1616 # crypto/des
1617 CRYPTO_DES= f_cbc.o f_cksum.o f_parity.o weak_key.o d3_cbc.o ef_crypto.o

1619 CRYPTO_DK= checksum.o derive.o dk_decrypt.o dk_encrypt.o

1621 CRYPTO_ARCFOUR= k5_arcfour.o

1623 # crypto/enc_provider
1624 CRYPTO_ENC= des.o des3.o arcfour_provider.o aes_provider.o

1626 # crypto/hash_provider
1627 CRYPTO_HASH= hash_kef_generic.o hash_kmd5.o hash_crc32.o hash_kshal.o

1629 # crypto/keyhash_provider
1630 CRYPTO_KEYHASH= descbc.o k5_kmd5des.o k_hmac_md5.o

1632 # crypto/crc32
1633 CRYPTO_CRC32= crc32.o

1635 # crypto/old
1636 CRYPTO_OLD= old_decrypt.o old_encrypt.o

1638 # crypto/raw
1639 CRYPTO_RAW= raw_decrypt.o raw_encrypt.o

1641 K5_KRB= kfree.o copy_key.o \
1642         parse.o init_ctx.o \
1643         ser_adata.o ser_addr.o \
1644         ser_auth.o ser_cksum.o \
1645         ser_key.o ser_princ.o \

```

```

1646         serialize.o unparse.o \
1647         ser_actx.o

1649 K5_OS= timeofday.o toffset.o \
1650        init_os_ctx.o c_ustime.o

1652 SEAL=
1653 # EXPORT DELETE START
1654 SEAL= seal.o unseal.o
1655 # EXPORT DELETE END

1657 MECH= delete_sec_context.o \
1658        import_sec_context.o \
1659        gssapi_krb5.o \
1660        k5seal.o k5unseal.o k5sealv3.o \
1661        ser_sctx.o \
1662        sign.o \
1663        util_crypt.o \
1664        util_validate.o util_ordering.o \
1665        util_seqnum.o util_set.o util_seed.o \
1666        wrap_size_limit.o verify.o

1670 MECH_GEN= util_token.o

1673 KGSS_KRB5_OBJS += krb5mech.o \
1674                 $(MECH) $(SEAL) $(MECH_GEN) \
1675                 $(CRYPTO) $(CRYPTO_DES) $(CRYPTO_DK) $(CRYPTO_ARCFOUR) \
1676                 $(CRYPTO_ENC) $(CRYPTO_HASH) \
1677                 $(CRYPTO_KEYHASH) $(CRYPTO_CRC32) \
1678                 $(CRYPTO_OLD) \
1679                 $(CRYPTO_RAW) $(K5_KRB) $(K5_OS)

1681 DES_OBJS += des_crypt.o des_impl.o des_ks.o des_soft.o

1683 DLBOOT_OBJS += bootparam_xdr.o nfs_dlinet.o scan.o

1685 KRTLD_OBJS += kobj_bootflags.o getoptstr.o \
1686              kobj.o kobj_kdi.o kobj_lm.o kobj_subr.o

1688 MOD_OBJS += modctl.o modsubr.o modsysfile.o modconf.o modhash.o

1690 STRPLUMB_OBJS += strplumb.o

1692 CPR_OBJS += cpr_driver.o cpr_dump.o \
1693            cpr_main.o cpr_misc.o cpr_mod.o cpr_stat.o \
1694            cpr_uthread.o

1696 PROF_OBJS += prf.o

1698 SE_OBJS += se_driver.o

1700 SYSACCT_OBJS += acct.o

1702 ACCTCTL_OBJS += acctctl.o

1704 EXACCTSYS_OBJS += exacctsys.o

1706 KAIQ_OBJS += aio.o

1708 PCMCIA_OBJS += pcmcia.o cs.o cis.o cis_callout.o cis_handlers.o cis_params.o

1710 BUSRA_OBJS += busra.o

```



```

1712 PCS_OBJS += pcs.o
1714 PCAN_OBJS += pcan.o
1716 PCATA_OBJS += pcide.o pcdisk.o pclabel.o pcata.o
1718 PCSER_OBJS += pcser.o pcser_cis.o
1720 PCWL_OBJS += pcwl.o
1722 PSET_OBJS += pset.o
1724 OHCI_OBJS += ohci.o ohci_hub.o ohci_polled.o
1726 UHCI_OBJS += uhci.o uhciutil.o uhcitgt.o uhcihub.o uhcipolled.o
1728 EHCI_OBJS += ehci.o ehci_hub.o ehci_xfer.o ehci_intr.o ehci_util.o ehci_polled.o
1730 HUBD_OBJS += hubd.o
1732 USB_MID_OBJS += usb_mid.o
1734 USB_IA_OBJS += usb_ia.o
1736 UWBA_OBJS += uwba.o uwbai.o
1738 SCSA2USB_OBJS += scsa2usb.o usb_ms_bulkonly.o usb_ms_cbi.o
1740 HWAHC_OBJS += hwahc.o hwahc_util.o
1742 WUSB_DF_OBJS += wusb_df.o
1743 WUSB_FWMOD_OBJS += wusb_fwmod.o
1745 IPF_OBJS += ip_fil_solaris.o fil.o solaris.o ip_state.o ip_frag.o ip_nat.o \
1746 ip_proxy.o ip_auth.o ip_pool.o ip_htable.o ip_lookup.o \
1747 ip_log.o misc.o ip_compat.o ip_nat6.o drand48.o
1749 IBD_OBJS += ibd.o ibd_cm.o
1751 EIBNX_OBJS += enx_main.o enx_hdlrs.o enx_ibt.o enx_log.o enx_fip.o \
1752 enx_misc.o enx_q.o enx_ctl.o
1754 EOIB_OBJS += eib_adm.o eib_chan.o eib_cmn.o eib_ctl.o eib_data.o \
1755 eib_fip.o eib_ibt.o eib_log.o eib_mac.o eib_main.o \
1756 eib_rsrc.o eib_svc.o eib_vnic.o
1758 DLPSTUB_OBJS += dlpstub.o
1760 SDP_OBJS += sdpddi.o
1762 TRILL_OBJS += trill.o
1764 CTF_OBJS += ctf_create.o ctf_decl.o ctf_error.o ctf_hash.o ctf_labels.o \
1765 ctf_lookup.o ctf_open.o ctf_types.o ctf_util.o ctf_subr.o ctf_mod.o
1767 SMBIOS_OBJS += smb_error.o smb_info.o smb_open.o smb_subr.o smb_dev.o
1769 RPCIB_OBJS += rpcib.o
1771 KMDB_OBJS += kdrv.o
1773 AFE_OBJS += afe.o
1775 BGE_OBJS += bge_main2.o bge_chip2.o bge_kstats.o bge_log.o bge_ndd.o \
1776 bge_atomic.o bge_mii.o bge_send.o bge_rcv2.o bge_mii_5906.o

```

```

1778 DMFE_OBJS += dmfe_log.o dmfe_main.o dmfe_mii.o
1780 EFE_OBJS += efe.o
1782 ELXL_OBJS += elxl.o
1784 HME_OBJS += hme.o
1786 IXGB_OBJS += ixgb.o ixgb_atomic.o ixgb_chip.o ixgb_gld.o ixgb_kstats.o \
1787 ixgb_log.o ixgb_ndd.o ixgb_rx.o ixgb_tx.o ixgb_xmii.o
1789 NGE_OBJS += nge_main.o nge_atomic.o nge_chip.o nge_ndd.o nge_kstats.o \
1790 nge_log.o nge_rx.o nge_tx.o nge_xmii.o
1792 PCN_OBJS += pcn.o
1794 RGE_OBJS += rge_main.o rge_chip.o rge_ndd.o rge_kstats.o rge_log.o rge_rxtx.o
1796 URTW_OBJS += urtw.o
1798 ARN_OBJS += arn_hw.o arn_eeprom.o arn_mac.o arn_calib.o arn_ani.o arn_phy.o arn_
1799 arn_main.o arn_rcv.o arn_xmit.o arn_rc.o
1801 ATH_OBJS += ath_aux.o ath_main.o ath_osdep.o ath_rate.o
1803 ATU_OBJS += atu.o
1805 IPW_OBJS += ipw2100_hw.o ipw2100.o
1807 IWI_OBJS += ipw2200_hw.o ipw2200.o
1809 IWH_OBJS += iwh.o
1811 IWK_OBJS += iwk2.o
1813 IWP_OBJS += iwp.o
1815 MWL_OBJS += mwl.o
1817 MWLFW_OBJS += mwlfw_mode.o
1819 WPI_OBJS += wpi.o
1821 RAL_OBJS += rt2560.o ral_rate.o
1823 RUM_OBJS += rum.o
1825 RWD_OBJS += rt2661.o
1827 RWN_OBJS += rt2860.o
1829 UATH_OBJS += uath.o
1831 UATHFW_OBJS += uathfw_mod.o
1833 URAL_OBJS += ural.o
1835 RTW_OBJS += rtw.o smc93cx6.o rtwphy.o rtwphyio.o
1837 ZYD_OBJS += zyd.o zyd_usb.o zyd_hw.o zyd_fw.o
1839 MXFE_OBJS += mxfe.o
1841 MPTSAS_OBJS += mptsas.o mptsas_impl.o mptsas_init.o mptsas_raid.o mptsas_smhba.o
1843 SFE_OBJS += sfe.o sfe_util.o

```

```

1845 BFE_OBJS += bfe.o
1847 BRIDGE_OBJS += bridge.o
1849 IDM_SHARED_OBJS += base64.o
1851 IDM_OBJS += $(IDM_SHARED_OBJS) \
1852     idm.o idm_impl.o idm_text.o idm_conn_sm.o idm_so.o
1854 VR_OBJS += vr.o
1856 ATGE_OBJS += atge_main.o atge_lle.o atge_mii.o atge_ll.o atge_llc.o
1858 YGE_OBJS = yge.o
1860 #
1861 #     Build up defines and paths.
1862 #
1863 LINT_DEFS     += -Dunix
1865 #
1866 #     This duality can be removed when the native and target compilers
1867 #     are the same (or at least recognize the same command line syntax!)
1868 #     It is a bug in the current compilation system that the assembler
1869 #     can't process the -Y I, flag.
1870 #
1871 NATIVE_INC_PATH += $(INC_PATH) $(CCYFLAG)$(UTSBASE)/common
1872 AS_INC_PATH     += $(INC_PATH) -I$(UTSBASE)/common
1873 INCLUDE_PATH   += $(INC_PATH) $(CCYFLAG)$(UTSBASE)/common
1875 PCIEB_OBJS += pcieb.o
1877 #     Chelsio N110 10G NIC driver module
1878 #
1879 CH_OBJS = ch.o glue.o pe.o sge.o
1881 CH_COM_OBJS = ch_mac.o ch_subr.o csapi.o espi.o ixfl1010.o mc3.o mc4.o mc5.o \
1882     mv88elxxx.o mv88x201x.o my3126.o pm3393.o tp.o ulp.o \
1883     vsc7321.o vsc7326.o xpak.o
1885 #
1886 #     Chelsio Terminator 4 10G NIC nexus driver module
1887 #
1888 CXGBE_FW_OBJS = t4_fw.o t4_cfg.o
1889 CXGBE_COM_OBJS = t4_hw.o common.o
1890 CXGBE_NEX_OBJS = t4_nexus.o t4_sge.o t4_mac.o t4_ioctl.o shared.o \
1891     t4_l2t.o adapter.o osdep.o
1893 #
1894 #     Chelsio Terminator 4 10G NIC driver module
1895 #
1896 CXGBE_OBJS = cxgbe.o
1898 #
1899 #     PCI strings file
1900 #
1901 PCI_STRING_OBJS = pci_strings.o
1903 NET_DACF_OBJS += net_dacf.o
1905 #
1906 #     Xframe 10G NIC driver module
1907 #
1908 XGE_OBJS = xge.o xgell.o

```

```

1910 XGE_HAL_OBJS = xgehal-channel.o xgehal-fifo.o xgehal-ring.o xgehal-config.o \
1911     xgehal-driver.o xgehal-mm.o xgehal-stats.o xgehal-device.o \
1912     xge-queue.o xgehal-mgmt.o xgehal-mgmtaux.o
1914 #
1915 #     e1000g module
1916 #
1917 E1000G_OBJS += e1000_80003es2lan.o e1000_82540.o e1000_82541.o e1000_82542.o \
1918     e1000_82543.o e1000_82571.o e1000_api.o e1000_ich8lan.o \
1919     e1000_mac.o e1000_manage.o e1000_nvmm.o e1000_osdep.o \
1920     e1000_phy.o e1000g_debug.o e1000g_main.o e1000g_alloc.o \
1921     e1000g_tx.o e1000g_rx.o e1000g_stat.o
1923 #
1924 #     Intel 82575 1G NIC driver module
1925 #
1926 IGB_OBJS = igb_82575.o igb_api.o igb_mac.o igb_manage.o \
1927     igb_nvmm.o igb_osdep.o igb_phy.o igb_buf.o \
1928     igb_debug.o igb_gld.o igb_log.o igb_main.o \
1929     igb_rx.o igb_stat.o igb_tx.o
1931 #
1932 #     Intel Pro/100 NIC driver module
1933 #
1934 IPRB_OBJS = iprb.o
1936 #
1937 #     Intel 10GbE PCIE NIC driver module
1938 #
1939 IXGBE_OBJS = ixgbe_82598.o ixgbe_82599.o ixgbe_api.o \
1940     ixgbe_common.o ixgbe_phy.o \
1941     ixgbe_buf.o ixgbe_debug.o ixgbe_gld.o \
1942     ixgbe_log.o ixgbe_main.o \
1943     ixgbe_osdep.o ixgbe_rx.o ixgbe_stat.o \
1944     ixgbe_tx.o ixgbe_x540.o ixgbe_mbx.o
1946 #
1947 #     NIU 10G/1G driver module
1948 #
1949 NXGE_OBJS = nxge_mac.o nxge_ipp.o nxge_rxdma.o \
1950     nxge_txdma.o nxge_txc.o nxge_main.o \
1951     nxge_hw.o nxge_fzc.o nxge_virtual.o \
1952     nxge_send.o nxge_classify.o nxge_fflp.o \
1953     nxge_fflp_hash.o nxge_ndd.o nxge_kstats.o \
1954     nxge_zcp.o nxge_fm.o nxge_espc.o nxge_hv.o \
1955     nxge_hio.o nxge_hio_guest.o nxge_intr.o
1957 NXGE_NPI_OBJS = \
1958     npi.o npi_mac.o npi_ipp.o \
1959     npi_txdma.o npi_rxdma.o npi_txc.o \
1960     npi_zcp.o npi_espc.o npi_fflp.o \
1961     npi_vir.o
1963 NXGE_HCALL_OBJS = \
1964     nxge_hcall.o
1966 #
1967 # Virtio modules
1968 #
1970 # Virtio core
1971 VIRTIO_OBJS = virtio.o
1973 # Virtio block driver
1974 VIOBLK_OBJS = vioblk.o

```

new/usr/src/uts/common/Makefile.files

31

```
1976 #
1977 #     kiconv modules
1978 #
1979 KICONV_EMEA_OBJS += kiconv_emea.o

1981 KICONV_JA_OBJS += kiconv_ja.o

1983 KICONV_KO_OBJS += kiconv_cck_common.o kiconv_ko.o

1985 KICONV_SC_OBJS += kiconv_cck_common.o kiconv_sc.o

1987 KICONV_TC_OBJS += kiconv_cck_common.o kiconv_tc.o

1989 #
1990 #     AAC module
1991 #
1992 AAC_OBJS = aac.o aac_ioctl.o

1994 #
1995 #     sdcard modules
1996 #
1997 SDA_OBJS =     sda_cmd.o sda_host.o sda_init.o sda_mem.o sda_mod.o sda_slot.o
1998 SDHOST_OBJS = sdhost.o

2000 #
2001 #     hxge 10G driver module
2002 #
2003 HXGE_OBJS =     hxge_main.o hxge_vmac.o hxge_send.o           \
2004                hxge_txdma.o hxge_rxdma.o hxge_virtual.o     \
2005                hxge_fm.o hxge_fzc.o hxge_hw.o hxge_kstats.o \
2006                hxge_ndd.o hxge_pfc.o                       \
2007                hpi.o hpi_vmac.o hpi_rxdma.o hpi_txdma.o     \
2008                hpi_vir.o hpi_pfc.o

2010 #
2011 #     MEGARAID_SAS module
2012 #
2013 MEGA_SAS_OBJS = megaraid_sas.o

2015 #
2016 #     MR_SAS module
2017 #
2018 MR_SAS_OBJS = ld_pd_map.o mr_sas.o mr_sas_tbolt.o mr_sas_list.o

2020 #
2021 #     ISCSI_INITIATOR module
2022 #
2023 ISCSI_INITIATOR_OBJS = chap.o iscsi_io.o iscsi_thread.o     \
2024                       iscsi_ioctl.o iscsid.o iscsi.o        \
2025                       iscsi_login.o isns_client.o iscsiAuthClient.o \
2026                       iscsi_lun.o iscsiAuthClientGlue.o     \
2027                       iscsi_net.o nvfile.o iscsi_cmd.o      \
2028                       iscsi_queue.o persistent.o iscsi_conn.o \
2029                       iscsi_sess.o radius_auth.o iscsi_crc.o \
2030                       iscsi_stats.o radius_packet.o iscsi_doorclt.o \
2031                       iscsi_targetparam.o utils.o kifconf.o

2033 #
2034 #     ntxn 10Gb/1Gb NIC driver module
2035 #
2036 NTXN_OBJS =     unm_nic_init.o unm_gem.o unm_nic_hw.o unm_ndd.o \
2037                unm_nic_main.o unm_nic_isr.o unm_nic_ctx.o niu.o

2039 #
2040 #     Myricom 10Gb NIC driver module
2041 #
```

new/usr/src/uts/common/Makefile.files

32

```
2042 MYRI10GE_OBJS = myri10ge.o myri10ge_lro.o

2044 #     nulldriver module
2045 #
2046 NULLDRIVER_OBJS =     nulldriver.o

2048 TPM_OBJS =     tpm.o tpm_hcall.o
```

```

*****
73704 Tue Sep 3 18:09:40 2013
new/usr/src/uts/common/Makefile.rules
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 #
22 #
23 # Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright (c) 2012 Nexenta Systems, Inc. All rights reserved.
25 #
26 #
27 #
28 # uts/common/Makefile.rules
29 #
30 # This Makefile defines all the file build rules for the directory
31 # uts/common and its children. These are the source files which may
32 # be considered common to all SunOS systems.
33 #
34 # The following two-level ordering must be maintained in this file.
35 # Lines are sorted first in order of decreasing specificity based on
36 # the first directory component. That is, sun4u rules come before
37 # sparc rules come before common rules.
38 #
39 # Lines whose initial directory components are equal are sorted
40 # alphabetically by the remaining components.
41 #
42 #
43 # Section 1a: C objects build rules
44 #
45 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/aes/%.c
46 $(COMPILE.c) -o $@ $<
47 $(CTFCONVERT_O)
48 #
49 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/arcfour/%.c
50 $(COMPILE.c) -o $@ $<
51 $(CTFCONVERT_O)
52 #
53 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/blowfish/%.c
54 $(COMPILE.c) -o $@ $<
55 $(CTFCONVERT_O)
56 #
57 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/ecc/%.c
58 $(COMPILE.c) -o $@ $<
59 $(CTFCONVERT_O)
60 #
61 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/modes/%.c

```

```

62 $(COMPILE.c) -o $@ $<
63 $(CTFCONVERT_O)
64 #
65 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/padding/%.c
66 $(COMPILE.c) -o $@ $<
67 $(CTFCONVERT_O)
68 #
69 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/rng/%.c
70 $(COMPILE.c) -o $@ $<
71 $(CTFCONVERT_O)
72 #
73 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/rsa/%.c
74 $(COMPILE.c) -o $@ $<
75 $(CTFCONVERT_O)
76 #
77 $(OBJSDIR)/%.o: $(COMMONBASE)/bignum/%.c
78 $(COMPILE.c) -o $@ $<
79 $(CTFCONVERT_O)
80 #
81 $(OBJSDIR)/%.o: $(UTSBASE)/common/bignum/%.c
82 $(COMPILE.c) -o $@ $<
83 $(CTFCONVERT_O)
84 #
85 $(OBJSDIR)/%.o: $(COMMONBASE)/mpi/%.c
86 $(COMPILE.c) -o $@ $<
87 $(CTFCONVERT_O)
88 #
89 $(OBJSDIR)/%.o: $(COMMONBASE)/acl/%.c
90 $(COMPILE.c) -o $@ $<
91 $(CTFCONVERT_O)
92 #
93 $(OBJSDIR)/%.o: $(COMMONBASE)/avl/%.c
94 $(COMPILE.c) -o $@ $<
95 $(CTFCONVERT_O)
96 #
97 $(OBJSDIR)/%.o: $(COMMONBASE)/ucode/%.c
98 $(COMPILE.c) -o $@ $<
99 $(CTFCONVERT_O)
100 #
101 $(OBJSDIR)/%.o: $(UTSBASE)/common/brand/sn1/%.c
102 $(COMPILE.c) -o $@ $<
103 $(CTFCONVERT_O)
104 #
105 $(OBJSDIR)/%.o: $(UTSBASE)/common/brand/solaris10/%.c
106 $(COMPILE.c) -o $@ $<
107 $(CTFCONVERT_O)
108 #
109 $(OBJSDIR)/%.o: $(UTSBASE)/common/c2/%.c
110 $(COMPILE.c) -o $@ $<
111 $(CTFCONVERT_O)
112 #
113 $(OBJSDIR)/%.o: $(UTSBASE)/common/conf/%.c
114 $(COMPILE.c) -o $@ $<
115 $(CTFCONVERT_O)
116 #
117 $(OBJSDIR)/%.o: $(UTSBASE)/common/contract/%.c
118 $(COMPILE.c) -o $@ $<
119 $(CTFCONVERT_O)
120 #
121 $(OBJSDIR)/%.o: $(UTSBASE)/common/cpr/%.c
122 $(COMPILE.c) -o $@ $<
123 $(CTFCONVERT_O)
124 #
125 $(OBJSDIR)/%.o: $(UTSBASE)/common/ctf/%.c
126 $(COMPILE.c) -o $@ $<
127 $(CTFCONVERT_O)

```

```

129 $(OBJSDIR)/%.o: $(COMMONBASE)/ctf/%.c
130 $(COMPILE.c) -o $@ $<
131 $(CTFCONVERT_O)

133 $(OBJSDIR)/%.o: $(COMMONBASE)/crypto/des/%.c
134 $(COMPILE.c) -o $@ $<
135 $(CTFCONVERT_O)

137 $(OBJSDIR)/%.o: $(COMMONBASE)/smbios/%.c
138 $(COMPILE.c) -o $@ $<
139 $(CTFCONVERT_O)

141 $(OBJSDIR)/%.o: $(UTSBASE)/common/des/%.c
142 $(COMPILE.c) -o $@ $<
143 $(CTFCONVERT_O)

145 $(OBJSDIR)/%.o: $(UTSBASE)/common/crypto/api/%.c
146 $(COMPILE.c) -o $@ $<
147 $(CTFCONVERT_O)

149 $(OBJSDIR)/%.o: $(UTSBASE)/common/crypto/core/%.c
150 $(COMPILE.c) -o $@ $<
151 $(CTFCONVERT_O)

153 $(OBJSDIR)/%.o: $(UTSBASE)/common/crypto/io/%.c
154 $(COMPILE.c) -o $@ $<
155 $(CTFCONVERT_O)

157 $(OBJSDIR)/%.o: $(UTSBASE)/common/crypto/spi/%.c
158 $(COMPILE.c) -o $@ $<
159 $(CTFCONVERT_O)

161 $(OBJSDIR)/%.o: $(COMMONBASE)/pci/%.c
162 $(COMPILE.c) -o $@ $<
163 $(CTFCONVERT_O)

165 $(OBJSDIR)/%.o: $(COMMONBASE)/devid/%.c
166 $(COMPILE.c) -o $@ $<
167 $(CTFCONVERT_O)

169 $(OBJSDIR)/%.o: $(UTSBASE)/common/disp/%.c
170 $(COMPILE.c) -o $@ $<
171 $(CTFCONVERT_O)

173 $(OBJSDIR)/%.o: $(UTSBASE)/common/dtrace/%.c
174 $(COMPILE.c) -o $@ $<
175 $(CTFCONVERT_O)

177 $(OBJSDIR)/%.o: $(COMMONBASE)/exacct/%.c
178 $(COMPILE.c) -o $@ $<
179 $(CTFCONVERT_O)

181 $(OBJSDIR)/%.o: $(UTSBASE)/common/exec/aout/%.c
182 $(COMPILE.c) -o $@ $<
183 $(CTFCONVERT_O)

185 $(OBJSDIR)/%.o: $(UTSBASE)/common/exec/elf/%.c
186 $(COMPILE.c) -o $@ $<
187 $(CTFCONVERT_O)

189 $(OBJSDIR)/%.o: $(UTSBASE)/common/exec/intp/%.c
190 $(COMPILE.c) -o $@ $<
191 $(CTFCONVERT_O)

193 $(OBJSDIR)/%.o: $(UTSBASE)/common/exec/shbin/%.c

```

```

194 $(COMPILE.c) -o $@ $<
195 $(CTFCONVERT_O)

197 $(OBJSDIR)/%.o: $(UTSBASE)/common/exec/java/%.c
198 $(COMPILE.c) -o $@ $<
199 $(CTFCONVERT_O)

201 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/%.c
202 $(COMPILE.c) -o $@ $<
203 $(CTFCONVERT_O)

205 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/autofs/%.c
206 $(COMPILE.c) -o $@ $<
207 $(CTFCONVERT_O)

209 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/cacheofs/%.c
210 $(COMPILE.c) -o $@ $<
211 $(CTFCONVERT_O)

213 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/dcfs/%.c
214 $(COMPILE.c) -o $@ $<
215 $(CTFCONVERT_O)

217 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/devfs/%.c
218 $(COMPILE.c) -o $@ $<
219 $(CTFCONVERT_O)

221 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/ctfs/%.c
222 $(COMPILE.c) -o $@ $<
223 $(CTFCONVERT_O)

225 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/doorfs/%.c
226 $(COMPILE.c) -o $@ $<
227 $(CTFCONVERT_O)

229 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/dev/%.c
230 $(COMPILE.c) -o $@ $<
231 $(CTFCONVERT_O)

233 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/fd/%.c
234 $(COMPILE.c) -o $@ $<
235 $(CTFCONVERT_O)

237 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/fifofs/%.c
238 $(COMPILE.c) -o $@ $<
239 $(CTFCONVERT_O)

241 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/hsfs/%.c
242 $(COMPILE.c) -o $@ $<
243 $(CTFCONVERT_O)

245 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/lofs/%.c
246 $(COMPILE.c) -o $@ $<
247 $(CTFCONVERT_O)

249 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/mntfs/%.c
250 $(COMPILE.c) -o $@ $<
251 $(CTFCONVERT_O)

253 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/namefs/%.c
254 $(COMPILE.c) -o $@ $<
255 $(CTFCONVERT_O)

257 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/nfs/%.c
258 $(COMPILE.c) -o $@ $<
259 $(CTFCONVERT_O)

```

```

261 $(OBJSDIR)/%.o: $(COMMONBASE)/smbsrv/%.c
262 $(COMPILE.c) -o $@ $<
263 $(CTFCONVERT_O)

265 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/smbsrv/%.c
266 $(COMPILE.c) -o $@ $<
267 $(CTFCONVERT_O)

269 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/objfs/%.c
270 $(COMPILE.c) -o $@ $<
271 $(CTFCONVERT_O)

273 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/pcfefs/%.c
274 $(COMPILE.c) -o $@ $<
275 $(CTFCONVERT_O)

277 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/portfs/%.c
278 $(COMPILE.c) -o $@ $<
279 $(CTFCONVERT_O)

281 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/proc/%.c
282 $(COMPILE.c) -o $@ $<
283 $(CTFCONVERT_O)

285 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/sharefs/%.c
286 $(COMPILE.c) -o $@ $<
287 $(CTFCONVERT_O)

289 $(OBJSDIR)/%.o: $(COMMONBASE)/smbclnt/%.c
290 $(COMPILE.c) -o $@ $<
291 $(CTFCONVERT_O)

293 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/smbclnt/net smb/%.c
294 $(COMPILE.c) -o $@ $<
295 $(CTFCONVERT_O)

297 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/smbclnt/smbfs/%.c
298 $(COMPILE.c) -o $@ $<
299 $(CTFCONVERT_O)

301 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/sockfs/%.c
302 $(COMPILE.c) -o $@ $<
303 $(CTFCONVERT_O)

305 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/specfs/%.c
306 $(COMPILE.c) -o $@ $<
307 $(CTFCONVERT_O)

309 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/swapfs/%.c
310 $(COMPILE.c) -o $@ $<
311 $(CTFCONVERT_O)

313 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/tmpfs/%.c
314 $(COMPILE.c) -o $@ $<
315 $(CTFCONVERT_O)

317 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/udfs/%.c
318 $(COMPILE.c) -o $@ $<
319 $(CTFCONVERT_O)

321 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/ufs/%.c
322 $(COMPILE.c) -o $@ $<
323 $(CTFCONVERT_O)

325 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/vscan/%.c

```

```

326 $(COMPILE.c) -o $@ $<
327 $(CTFCONVERT_O)

329 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/zfs/%.c
330 $(COMPILE.c) -o $@ $<
331 $(CTFCONVERT_O)

333 $(OBJSDIR)/%.o: $(UTSBASE)/common/fs/zut/%.c
334 $(COMPILE.c) -o $@ $<
335 $(CTFCONVERT_O)

337 $(OBJSDIR)/%.o: $(COMMONBASE)/xattr/%.c
338 $(COMPILE.c) -o $@ $<
339 $(CTFCONVERT_O)

341 $(OBJSDIR)/%.o: $(COMMONBASE)/zfs/%.c
342 $(COMPILE.c) -o $@ $<
343 $(CTFCONVERT_O)

345 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/pmcs/%.c
346 $(COMPILE.c) -o $@ $<
347 $(CTFCONVERT_O)

349 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/pmcs/%.bin
350 $(COMPILE.b) -o $@ $<
351 $(CTFCONVERT_O)

353 $(OBJSDIR)/%.o: $(COMMONBASE)/fsreparse/%.c
354 $(COMPILE.c) -o $@ $<
355 $(CTFCONVERT_O)

357 KMECHKRB5_BASE=$(UTSBASE)/common/gssapi/mechs/krb5

359 KGSSDFLAGS=-I $(UTSBASE)/common/gssapi/include

361 # Note, KRB5_DEFS can be assigned various preprocessor flags,
362 # typically -D defines on the make invocation. The standard compiler
363 # flags will not be overwritten.
364 KGSSDFLAGS += $(KRB5_DEFS)

366 $(OBJSDIR)/%.o: $(UTSBASE)/common/gssapi/%.c
367 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
368 $(CTFCONVERT_O)

370 $(OBJSDIR)/%.o: $(UTSBASE)/common/gssapi/mechs/dummy/%.c
371 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
372 $(CTFCONVERT_O)

374 $(OBJSDIR)/%.o: $(KMECHKRB5_BASE)/%.c
375 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
376 $(CTFCONVERT_O)

378 $(OBJSDIR)/%.o: $(KMECHKRB5_BASE)/crypto/%.c
379 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
380 $(CTFCONVERT_O)

382 $(OBJSDIR)/%.o: $(KMECHKRB5_BASE)/crypto/des/%.c
383 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
384 $(CTFCONVERT_O)

386 $(OBJSDIR)/%.o: $(KMECHKRB5_BASE)/crypto/arcfour/%.c
387 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
388 $(CTFCONVERT_O)

390 $(OBJSDIR)/%.o: $(KMECHKRB5_BASE)/crypto/dk/%.c
391 $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<

```

```

392      $(CTFCONVERT_O)

394 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/crypto/enc_provider/%.c
395     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
396     $(CTFCONVERT_O)

398 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/crypto/hash_provider/%.c
399     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
400     $(CTFCONVERT_O)

402 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/crypto/keyhash_provider/%.c
403     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
404     $(CTFCONVERT_O)

406 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/crypto/raw/%.c
407     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
408     $(CTFCONVERT_O)

410 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/crypto/old/%.c
411     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
412     $(CTFCONVERT_O)

414 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/krb5/krb/%.c
415     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
416     $(CTFCONVERT_O)

418 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/krb5/os/%.c
419     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
420     $(CTFCONVERT_O)

422 $(OBJS_DIR)/ser_sctx.o := CPPFLAGS += -DPROVIDE_KERNEL_IMPORT=1

424 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/mech/%.c
425     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
426     $(CTFCONVERT_O)

428 $(OBJS_DIR)/%.o:                $(KMECHKRB5_BASE)/profile/%.c
429     $(COMPILE.c) $(KGSSDFLAGS) -o $@ $<
430     $(CTFCONVERT_O)

432 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ncall/%.c
433     $(COMPILE.c) -o $@ $<
434     $(CTFCONVERT_O)

436 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/dsw/%.c
437     $(COMPILE.c) -o $@ $<
438     $(CTFCONVERT_O)

440 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/nsctl/%.c
441     $(COMPILE.c) -o $@ $<
442     $(CTFCONVERT_O)

444 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/rdc/%.c
445     $(COMPILE.c) -o $@ $<
446     $(CTFCONVERT_O)

448 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/sdbc/%.c
449     $(COMPILE.c) -o $@ $<
450     $(CTFCONVERT_O)

452 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/solaris/%.c
453     $(COMPILE.c) -o $@ $<
454     $(CTFCONVERT_O)

456 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/sv/%.c
457     $(COMPILE.c) -o $@ $<

```

```

458      $(CTFCONVERT_O)

460 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/avs/ns/unistat/%.c
461     $(COMPILE.c) -o $@ $<
462     $(CTFCONVERT_O)

464 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/idmap/%.c
465     $(COMPILE.c) -o $@ $<
466     $(CTFCONVERT_O)

468 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/%.c
469     $(COMPILE.c) -o $@ $<
470     $(CTFCONVERT_O)

472 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/arp/%.c
473     $(COMPILE.c) -o $@ $<
474     $(CTFCONVERT_O)

476 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/ip/%.c
477     $(COMPILE.c) -o $@ $<
478     $(CTFCONVERT_O)

480 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/ipnet/%.c
481     $(COMPILE.c) -o $@ $<
482     $(CTFCONVERT_O)

484 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/iptun/%.c
485     $(COMPILE.c) -o $@ $<
486     $(CTFCONVERT_O)

488 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/kssl/%.c
489     $(COMPILE.c) -o $@ $<
490     $(CTFCONVERT_O)

492 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/sctp/%.c
493     $(COMPILE.c) -o $@ $<
494     $(CTFCONVERT_O)

496 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/tcp/%.c
497     $(COMPILE.c) -o $@ $<
498     $(CTFCONVERT_O)

500 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/ilb/%.c
501     $(COMPILE.c) -o $@ $<
502     $(CTFCONVERT_O)

504 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/ipf/%.c
505     $(COMPILE.c) -o $@ $<
506     $(CTFCONVERT_O)

508 $(OBJS_DIR)/%.o:                $(COMMONBASE)/net/patricia/%.c
509     $(COMPILE.c) -o $@ $<
510     $(CTFCONVERT_O)

512 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/udp/%.c
513     $(COMPILE.c) -o $@ $<
514     $(CTFCONVERT_O)

516 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/nca/%.c
517     $(COMPILE.c) -o $@ $<
518     $(CTFCONVERT_O)

520 $(OBJS_DIR)/%.o:                $(UTSBASE)/common/inet/sockmods/%.c
521     $(COMPILE.c) -o $@ $<
522     $(CTFCONVERT_O)

```

```

524 $(OBJSDIR)/%.o: $(UTSBASE)/common/inet/dlpistub/%.c
525 $(COMPILE.c) -o $@ $<
526 $(CTFCONVERT_O)

528 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/%.c
529 $(COMPILE.c) -o $@ $<
530 $(CTFCONVERT_O)

532 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/l394/%.c
533 $(COMPILE.c) -o $@ $<
534 $(CTFCONVERT_O)

536 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/l394/adapters/%.c
537 $(COMPILE.c) -o $@ $<
538 $(CTFCONVERT_O)

540 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/l394/targets/av1394/%.c
541 $(COMPILE.c) -o $@ $<
542 $(CTFCONVERT_O)

544 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/l394/targets/dcam1394/%.c
545 $(COMPILE.c) -o $@ $<
546 $(CTFCONVERT_O)

548 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/l394/targets/scsal394/%.c
549 $(COMPILE.c) -o $@ $<
550 $(CTFCONVERT_O)

552 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sbp2/%.c
553 $(COMPILE.c) -o $@ $<
554 $(CTFCONVERT_O)

556 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/aac/%.c
557 $(COMPILE.c) -o $@ $<
558 $(CTFCONVERT_O)

560 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/afe/%.c
561 $(COMPILE.c) -o $@ $<
562 $(CTFCONVERT_O)

564 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/atge/%.c
565 $(COMPILE.c) -o $@ $<
566 $(CTFCONVERT_O)

568 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/arn/%.c
569 $(COMPILE.c) -o $@ $<
570 $(CTFCONVERT_O)

572 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ath/%.c
573 $(COMPILE.c) -o $@ $<
574 $(CTFCONVERT_O)

576 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/atu/%.c
577 $(COMPILE.c) -o $@ $<
578 $(CTFCONVERT_O)

580 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/impl/%.c
581 $(COMPILE.c) -o $@ $<
582 $(CTFCONVERT_O)

584 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/ac97/%.c
585 $(COMPILE.c) -o $@ $<
586 $(CTFCONVERT_O)

588 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audioens/%.c
589 $(COMPILE.c) -o $@ $<

```

```

590 $(CTFCONVERT_O)

592 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audioemu10k/%.c
593 $(COMPILE.c) -o $@ $<
594 $(CTFCONVERT_O)

596 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audio1575/%.c
597 $(COMPILE.c) -o $@ $<
598 $(CTFCONVERT_O)

600 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audio810/%.c
601 $(COMPILE.c) -o $@ $<
602 $(CTFCONVERT_O)

604 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiocmi/%.c
605 $(COMPILE.c) -o $@ $<
606 $(CTFCONVERT_O)

608 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiocmihd/%.c
609 $(COMPILE.c) -o $@ $<
610 $(CTFCONVERT_O)

612 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiohd/%.c
613 $(COMPILE.c) -o $@ $<
614 $(CTFCONVERT_O)

616 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audioixp/%.c
617 $(COMPILE.c) -o $@ $<
618 $(CTFCONVERT_O)

620 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiols/%.c
621 $(COMPILE.c) -o $@ $<
622 $(CTFCONVERT_O)

624 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiopci/%.c
625 $(COMPILE.c) -o $@ $<
626 $(CTFCONVERT_O)

628 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiopl6x/%.c
629 $(COMPILE.c) -o $@ $<
630 $(CTFCONVERT_O)

632 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiosolo/%.c
633 $(COMPILE.c) -o $@ $<
634 $(CTFCONVERT_O)

636 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiots/%.c
637 $(COMPILE.c) -o $@ $<
638 $(CTFCONVERT_O)

640 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiovia823x/%.c
641 $(COMPILE.c) -o $@ $<
642 $(CTFCONVERT_O)

644 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/audio/drv/audiovia97/%.c
645 $(COMPILE.c) -o $@ $<
646 $(CTFCONVERT_O)

648 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/bfe/%.c
649 $(COMPILE.c) -o $@ $<
650 $(CTFCONVERT_O)

652 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/bge/%.c
653 $(COMPILE.c) -o $@ $<
654 $(CTFCONVERT_O)

```



```

656 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/blkdev/%.c
657 $(COMPILE.c) -o $@ $<
658 $(CTFCONVERT_O)

660 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/bpf/%.c
661 $(COMPILE.c) -o $@ $<
662 $(CTFCONVERT_O)

664 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cardbus/%.c
665 $(COMPILE.c) -o $@ $<
666 $(CTFCONVERT_O)

668 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/stmf/%.c
669 $(COMPILE.c) -o $@ $<
670 $(CTFCONVERT_O)

672 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/fct/%.c
673 $(COMPILE.c) -o $@ $<
674 $(CTFCONVERT_O)

676 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/qlt/%.c
677 $(COMPILE.c) -o $@ $<
678 $(CTFCONVERT_O)

680 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/srpt/%.c
681 $(COMPILE.c) -o $@ $<
682 $(CTFCONVERT_O)

684 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/fcoet/%.c
685 $(COMPILE.c) -o $@ $<
686 $(CTFCONVERT_O)

688 $(OBJSDIR)/%.o: $(COMMONBASE)/iscsit/%.c
689 $(COMPILE.c) -o $@ $<
690 $(CTFCONVERT_O)

692 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/iscsit/%.c
693 $(COMPILE.c) -o $@ $<
694 $(CTFCONVERT_O)

696 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/port/pppt/%.c
697 $(COMPILE.c) -o $@ $<
698 $(CTFCONVERT_O)

700 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/comstar/lu/stmf_sbd/%.c
701 $(COMPILE.c) -o $@ $<
702 $(CTFCONVERT_O)

704 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/dld/%.c
705 $(COMPILE.c) -o $@ $<
706 $(CTFCONVERT_O)

708 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/dls/%.c
709 $(COMPILE.c) -o $@ $<
710 $(CTFCONVERT_O)

712 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/dmfe/%.c
713 $(COMPILE.c) -o $@ $<
714 $(CTFCONVERT_O)

716 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/drm/%.c
717 $(COMPILE.c) -o $@ $<
718 $(CTFCONVERT_O)

720 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/efe/%.c
721 $(COMPILE.c) -o $@ $<

```

```

722 $(CTFCONVERT_O)

724 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/elxl/%.c
725 $(COMPILE.c) -o $@ $<
726 $(CTFCONVERT_O)

728 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fcoe/%.c
729 $(COMPILE.c) -o $@ $<
730 $(CTFCONVERT_O)

732 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fsd/%.c
733 $(COMPILE.c) -o $@ $<
734 $(CTFCONVERT_O)

736 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/hme/%.c
737 $(COMPILE.c) -o $@ $<
738 $(CTFCONVERT_O)

740 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pciex/%.c
741 $(COMPILE.c) -o $@ $<
742 $(CTFCONVERT_O)

744 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/hotplug/hpcsvc/%.c
745 $(COMPILE.c) -o $@ $<
746 $(CTFCONVERT_O)

748 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pciex/hotplug/%.c
749 $(COMPILE.c) -o $@ $<
750 $(CTFCONVERT_O)

752 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/hotplug/pcihp/%.c
753 $(COMPILE.c) -o $@ $<
754 $(CTFCONVERT_O)

756 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/rds/%.c
757 $(COMPILE.c) -o $@ $<
758 $(CTFCONVERT_O)

760 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/rdsv3/%.c
761 $(COMPILE.c) -o $@ $<
762 $(CTFCONVERT_O)

764 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/iser/%.c
765 $(COMPILE.c) -o $@ $<
766 $(CTFCONVERT_O)

768 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/ibd/%.c
769 $(COMPILE.c) -o $@ $<
770 $(CTFCONVERT_O)

772 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/eoib/%.c
773 $(COMPILE.c) -o $@ $<
774 $(CTFCONVERT_O)

776 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/of/sol_ofs/%.c
777 $(COMPILE.c) -o $@ $<
778 $(CTFCONVERT_O)

780 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/of/sol_ucma/%.c
781 $(COMPILE.c) -o $@ $<
782 $(CTFCONVERT_O)

784 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/of/sol_umad/%.c
785 $(COMPILE.c) -o $@ $<
786 $(CTFCONVERT_O)

```

```

788 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/of/sol_uverbs/%.
789 $(COMPILE.c) -o $@ $<
790 $(CTFCONVERT_O)

792 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/sdp/%.c
793 $(COMPILE.c) -o $@ $<
794 $(CTFCONVERT_O)

796 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/mgt/ibcm/%.c
797 $(COMPILE.c) -o $@ $<
798 $(CTFCONVERT_O)

800 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/mgt/ibdm/%.c
801 $(COMPILE.c) -o $@ $<
802 $(CTFCONVERT_O)

804 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/mgt/ibdma/%.c
805 $(COMPILE.c) -o $@ $<
806 $(CTFCONVERT_O)

808 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/mgt/ibmf/%.c
809 $(COMPILE.c) -o $@ $<
810 $(CTFCONVERT_O)

812 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/ibnex/%.c
813 $(COMPILE.c) -o $@ $<
814 $(CTFCONVERT_O)

816 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/ibtl/%.c
817 $(COMPILE.c) -o $@ $<
818 $(CTFCONVERT_O)

820 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/adapters/tavor/%.c
821 $(COMPILE.c) -o $@ $<
822 $(CTFCONVERT_O)

824 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/adapters/hermon/%.c
825 $(COMPILE.c) -o $@ $<
826 $(CTFCONVERT_O)

828 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ib/clients/daplt/%.c
829 $(COMPILE.c) -o $@ $<
830 $(CTFCONVERT_O)

832 $(OBJSDIR)/%.o: $(COMMONBASE)/iscsi/%.c
833 $(COMPILE.c) -o $@ $<
834 $(CTFCONVERT_O)

836 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/idm/%.c
837 $(COMPILE.c) -o $@ $<
838 $(CTFCONVERT_O)

840 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ipw/%.c
841 $(COMPILE.c) -o $@ $<
842 $(CTFCONVERT_O)

844 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/iwh/%.c
845 $(COMPILE.c) -o $@ $<
846 $(CTFCONVERT_O)

848 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/iwi/%.c
849 $(COMPILE.c) -o $@ $<
850 $(CTFCONVERT_O)

852 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/iwk/%.c
853 $(COMPILE.c) -o $@ $<

```

```

854 $(CTFCONVERT_O)

856 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/iwp/%.c
857 $(COMPILE.c) -o $@ $<
858 $(CTFCONVERT_O)

860 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/kb8042/%.c
861 $(COMPILE.c) -o $@ $<
862 $(CTFCONVERT_O)

864 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/kbtrans/%.c
865 $(COMPILE.c) -o $@ $<
866 $(CTFCONVERT_O)

868 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ksocket/%.c
869 $(COMPILE.c) -o $@ $<
870 $(CTFCONVERT_O)

872 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/aggr/%.c
873 $(COMPILE.c) -o $@ $<
874 $(CTFCONVERT_O)

876 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lp/%.c
877 $(COMPILE.c) -o $@ $<
878 $(CTFCONVERT_O)

880 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/hotspares/%.c
881 $(COMPILE.c) -o $@ $<
882 $(CTFCONVERT_O)

884 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/md/%.c
885 $(COMPILE.c) -o $@ $<
886 $(CTFCONVERT_O)

888 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/mirror/%.c
889 $(COMPILE.c) -o $@ $<
890 $(CTFCONVERT_O)

892 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/notify/%.c
893 $(COMPILE.c) -o $@ $<
894 $(CTFCONVERT_O)

896 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/raid/%.c
897 $(COMPILE.c) -o $@ $<
898 $(CTFCONVERT_O)

900 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/softpart/%.c
901 $(COMPILE.c) -o $@ $<
902 $(CTFCONVERT_O)

904 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/stripe/%.c
905 $(COMPILE.c) -o $@ $<
906 $(CTFCONVERT_O)

908 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/lvm/trans/%.c
909 $(COMPILE.c) -o $@ $<
910 $(CTFCONVERT_O)

912 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mac/%.c
913 $(COMPILE.c) -o $@ $<
914 $(CTFCONVERT_O)

916 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mac/plugins/%.c
917 $(COMPILE.c) -o $@ $<
918 $(CTFCONVERT_O)

```

```

920 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mega_sas/%.c
921     $(COMPILE.c) -o $@ $<
922     $(CTFCONVERT_O)

924 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mii/%.c
925     $(COMPILE.c) -o $@ $<
926     $(CTFCONVERT_O)

928 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mr_sas/%.c
929     $(COMPILE.c) -o $@ $<
930     $(CTFCONVERT_O)

932 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/mpt_sas/%.c
933     $(COMPILE.c) -o $@ $<
934     $(CTFCONVERT_O)

936 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mxfe/%.c
937     $(COMPILE.c) -o $@ $<
938     $(CTFCONVERT_O)

940 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mwl/%.c
941     $(COMPILE.c) -o $@ $<
942     $(CTFCONVERT_O)

944 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/mwl/mwl_fw/%.c
945     $(COMPILE.c) -o $@ $<
946     $(CTFCONVERT_O)

948 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/net80211/%.c
949     $(COMPILE.c) -o $@ $<
950     $(CTFCONVERT_O)

952 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/nge/%.c
953     $(COMPILE.c) -o $@ $<
954     $(CTFCONVERT_O)

956 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/nxge/%.c
957     $(COMPILE.c) -o $@ $<
958     $(CTFCONVERT_O)

960 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/nxge/mpi/%.c
961     $(COMPILE.c) -o $@ $<
962     $(CTFCONVERT_O)

964 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/nxge/%.s
965     $(COMPILE.s) -o $@ $<

967 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pci-ide/%.c
968     $(COMPILE.c) -o $@ $<
969     $(CTFCONVERT_O)

971 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pcmcia/%.c
972     $(COMPILE.c) -o $@ $<
973     $(CTFCONVERT_O)

975 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pcan/%.c
976     $(COMPILE.c) -o $@ $<
977     $(CTFCONVERT_O)

979 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pcn/%.c
980     $(COMPILE.c) -o $@ $<
981     $(CTFCONVERT_O)

983 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/pcwl/%.c
984     $(COMPILE.c) -o $@ $<
985     $(CTFCONVERT_O)

```

```

987 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ppp/sppp/%.c
988     $(COMPILE.c) -o $@ $<
989     $(CTFCONVERT_O)

991 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ppp/spppasyn/%.c
992     $(COMPILE.c) -o $@ $<
993     $(CTFCONVERT_O)

995 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ppp/sppptun/%.c
996     $(COMPILE.c) -o $@ $<
997     $(CTFCONVERT_O)

999 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ral/%.c
1000    $(COMPILE.c) -o $@ $<
1001    $(CTFCONVERT_O)

1003 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rge/%.c
1004    $(COMPILE.c) -o $@ $<
1005    $(CTFCONVERT_O)

1007 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rtls/%.c
1008    $(COMPILE.c) -o $@ $<
1009    $(CTFCONVERT_O)

1011 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rsm/%.c
1012    $(COMPILE.c) -o $@ $<
1013    $(CTFCONVERT_O)

1015 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rtw/%.c
1016    $(COMPILE.c) -o $@ $<
1017    $(CTFCONVERT_O)

1019 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rum/%.c
1020    $(COMPILE.c) -o $@ $<
1021    $(CTFCONVERT_O)

1023 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rwd/%.c
1024    $(COMPILE.c) -o $@ $<
1025    $(CTFCONVERT_O)

1027 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/rwn/%.c
1028    $(COMPILE.c) -o $@ $<
1029    $(CTFCONVERT_O)

1031 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sata/adapters/ahci/%.c
1032    $(COMPILE.c) -o $@ $<
1033    $(CTFCONVERT_O)

1035 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sata/adapters/nv_sata/%.c
1036    $(COMPILE.c) -o $@ $<
1037    $(CTFCONVERT_O)

1039 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sata/adapters/si3124/%.c
1040    $(COMPILE.c) -o $@ $<
1041    $(CTFCONVERT_O)

1043 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sata/impl/%.c
1044    $(COMPILE.c) -o $@ $<
1045    $(CTFCONVERT_O)

1047 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/conf/%.c
1048    $(COMPILE.c) -o $@ $<
1049    $(CTFCONVERT_O)

1051 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/impl/%.c

```

```

1052 $(COMPILE.c) -o $$@ $<
1053 $(CTFCONVERT_O)

1055 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/targets/%.c
1056 $(COMPILE.c) -o $$@ $<
1057 $(CTFCONVERT_O)

1059 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/%.c
1060 $(COMPILE.c) -o $$@ $<
1061 $(CTFCONVERT_O)

1063 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/blk2scsa/%.c
1064 $(COMPILE.c) -o $$@ $<
1065 $(CTFCONVERT_O)

1067 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/scsi_vhci/%.c
1068 $(COMPILE.c) -o $$@ $<
1069 $(CTFCONVERT_O)

1071 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/scsi_vhci/fop
1072 $(COMPILE.c) -o $$@ $<
1073 $(CTFCONVERT_O)

1075 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/ulp/%.c
1076 $(COMPILE.c) -o $$@ $<
1077 $(CTFCONVERT_O)

1079 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/impl/%.c
1080 $(COMPILE.c) -o $$@ $<
1081 $(CTFCONVERT_O)

1083 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/fca/qlc/%.c
1084 $(COMPILE.c) -o $$@ $<
1085 $(CTFCONVERT_O)

1087 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/fca/qlge/%.c
1088 $(COMPILE.c) -o $$@ $<
1089 $(CTFCONVERT_O)

1091 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/fca/emlxs/%.c
1092 $(COMPILE.c) -o $$@ $<
1093 $(CTFCONVERT_O)

1095 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/fca/oce/%.c
1096 $(COMPILE.c) -o $$@ $<
1097 $(CTFCONVERT_O)

1099 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/fibre-channel/fca/fcoei/%.c
1100 $(COMPILE.c) -o $$@ $<
1101 $(CTFCONVERT_O)

1103 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sdcard/adapters/sdhost/%.c
1104 $(COMPILE.c) -o $$@ $<
1105 $(CTFCONVERT_O)

1107 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sdcard/impl/%.c
1108 $(COMPILE.c) -o $$@ $<
1109 $(CTFCONVERT_O)

1111 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sdcard/targets/sdcard/%.c
1112 $(COMPILE.c) -o $$@ $<
1113 $(CTFCONVERT_O)

1115 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/sfe/%.c
1116 $(COMPILE.c) -o $$@ $<
1117 $(CTFCONVERT_O)

```

```

1119 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/simnet/%.c
1120 $(COMPILE.c) -o $$@ $<
1121 $(CTFCONVERT_O)

1123 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/softmac/%.c
1124 $(COMPILE.c) -o $$@ $<
1125 $(CTFCONVERT_O)

1127 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/uath/%.c
1128 $(COMPILE.c) -o $$@ $<
1129 $(CTFCONVERT_O)

1131 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/uath/uath_fw/%.c
1132 $(COMPILE.c) -o $$@ $<
1133 $(CTFCONVERT_O)

1135 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ural/%.c
1136 $(COMPILE.c) -o $$@ $<
1137 $(CTFCONVERT_O)

1139 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/urtw/%.c
1140 $(COMPILE.c) -o $$@ $<
1141 $(CTFCONVERT_O)

1143 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/audio/usb_ac/%.
1144 $(COMPILE.c) -o $$@ $<
1145 $(CTFCONVERT_O)

1147 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/audio/usb_as/%.
1148 $(COMPILE.c) -o $$@ $<
1149 $(CTFCONVERT_O)

1151 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/audio/usb_ah/%.
1152 $(COMPILE.c) -o $$@ $<
1153 $(CTFCONVERT_O)

1155 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbskel/%.c
1156 $(COMPILE.c) -o $$@ $<
1157 $(CTFCONVERT_O)

1159 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/video/usbvc/%.c
1160 $(COMPILE.c) -o $$@ $<
1161 $(CTFCONVERT_O)

1163 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/hwarc/%.c
1164 $(COMPILE.c) -o $$@ $<
1165 $(CTFCONVERT_O)

1167 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/hid/%.c
1168 $(COMPILE.c) -o $$@ $<
1169 $(CTFCONVERT_O)

1171 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/hidparser/%.c
1172 $(COMPILE.c) -o $$@ $<
1173 $(CTFCONVERT_O)

1175 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/printer/%.c
1176 $(COMPILE.c) -o $$@ $<
1177 $(CTFCONVERT_O)

1179 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbkbn/%.c
1180 $(COMPILE.c) -o $$@ $<
1181 $(CTFCONVERT_O)

1183 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usblms/%.c

```

```

1184 $(COMPILE.c) -o $@ $<
1185 $(CTFCONVERT_O)

1187 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbinput/usbwcm
1188 $(COMPILE.c) -o $@ $<
1189 $(CTFCONVERT_O)

1191 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/ugen/%.c
1192 $(COMPILE.c) -o $@ $<
1193 $(CTFCONVERT_O)

1195 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbser/%.c
1196 $(COMPILE.c) -o $@ $<
1197 $(CTFCONVERT_O)

1199 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbser/usbsacm/
1200 $(COMPILE.c) -o $@ $<
1201 $(CTFCONVERT_O)

1203 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbser/usbftdi/
1204 $(COMPILE.c) -o $@ $<
1205 $(CTFCONVERT_O)

1207 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbser/usbser_k
1208 $(COMPILE.c) -o $@ $<
1209 $(CTFCONVERT_O)

1211 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbser/usbsprl/
1212 $(COMPILE.c) -o $@ $<
1213 $(CTFCONVERT_O)

1215 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/wusb_df/%.c
1216 $(COMPILE.c) -o $@ $<
1217 $(CTFCONVERT_O)

1219 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/hwal480_fw/%.c
1220 $(COMPILE.c) -o $@ $<
1221 $(CTFCONVERT_O)

1223 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/wusb_ca/%.c
1224 $(COMPILE.c) -o $@ $<
1225 $(CTFCONVERT_O)

1227 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/clients/usbecm/%.c
1228 $(COMPILE.c) -o $@ $<
1229 $(CTFCONVERT_O)

1231 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/hcd/openhci/%.c
1232 $(COMPILE.c) -o $@ $<
1233 $(CTFCONVERT_O)

1235 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/hcd/ehci/%.c
1236 $(COMPILE.c) -o $@ $<
1237 $(CTFCONVERT_O)

1239 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/hcd/uhci/%.c
1240 $(COMPILE.c) -I../common -o $@ $<
1241 $(CTFCONVERT_O)

1243 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/hubd/%.c
1244 $(COMPILE.c) -o $@ $<
1245 $(CTFCONVERT_O)

1247 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/scsa2usb/%.c
1248 $(COMPILE.c) -o $@ $<
1249 $(CTFCONVERT_O)

```

```

1251 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/usb_mid/%.c
1252 $(COMPILE.c) -o $@ $<
1253 $(CTFCONVERT_O)

1255 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/usb_ia/%.c
1256 $(COMPILE.c) -o $@ $<
1257 $(CTFCONVERT_O)

1259 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/usba/%.c
1260 $(COMPILE.c) -o $@ $<
1261 $(CTFCONVERT_O)

1263 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/usba10/%.c
1264 $(COMPILE.c) -o $@ $<
1265 $(CTFCONVERT_O)

1267 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/usb/hwa/hwahc/%.c
1268 $(COMPILE.c) -o $@ $<
1269 $(CTFCONVERT_O)

1271 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/uwb/uwba/%.c
1272 $(COMPILE.c) -o $@ $<
1273 $(CTFCONVERT_O)

1275 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/vuidmice/%.c
1276 $(COMPILE.c) -o $@ $<
1277 $(CTFCONVERT_O)

1279 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/vnic/%.c
1280 $(COMPILE.c) -o $@ $<
1281 $(CTFCONVERT_O)

1283 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/wpi/%.c
1284 $(COMPILE.c) -o $@ $<
1285 $(CTFCONVERT_O)

1287 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/zyd/%.c
1288 $(COMPILE.c) -o $@ $<
1289 $(CTFCONVERT_O)

1291 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/chxge/com/%.c
1292 $(COMPILE.c) -o $@ $<
1293 $(CTFCONVERT_O)

1295 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/chxge/%.c
1296 $(COMPILE.c) -o $@ $<
1297 $(CTFCONVERT_O)

1299 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cxgbe/common/%.c
1300 $(COMPILE.c) -o $@ $<
1301 $(CTFCONVERT_O)

1303 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cxgbe/shared/%.c
1304 $(COMPILE.c) -o $@ $<
1305 $(CTFCONVERT_O)

1307 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cxgbe/firmware/%.c
1308 $(COMPILE.c) -o $@ $<
1309 $(CTFCONVERT_O)

1311 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cxgbe/t4nex/%.c
1312 $(COMPILE.c) -o $@ $<
1313 $(CTFCONVERT_O)

1315 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/cxgbe/cxgbe/%.c

```

```

1316 $(COMPILE.c) -o $$@ $<
1317 $(CTFCONVERT_O)

1319 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ixgb/%.c
1320 $(COMPILE.c) -o $$@ $<
1321 $(CTFCONVERT_O)

1323 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/xge/drv/%.c
1324 $(COMPILE.c) -o $$@ $<
1325 $(CTFCONVERT_O)

1327 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/xge/hal/xgehal/%.c
1328 $(COMPILE.c) -o $$@ $<
1329 $(CTFCONVERT_O)

1331 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/e1000g/%.c
1332 $(COMPILE.c) -o $$@ $<
1333 $(CTFCONVERT_O)

1335 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/igb/%.c
1336 $(COMPILE.c) -o $$@ $<
1337 $(CTFCONVERT_O)

1339 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/iprb/%.c
1340 $(COMPILE.c) -o $$@ $<
1341 $(CTFCONVERT_O)

1343 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ixgbe/%.c
1344 $(COMPILE.c) -o $$@ $<
1345 $(CTFCONVERT_O)

1347 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/ntxn/%.c
1348 $(COMPILE.c) -o $$@ $<
1349 $(CTFCONVERT_O)

1351 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/myril0ge/drv/%.c
1352 $(COMPILE.c) -o $$@ $<
1353 $(CTFCONVERT_O)

1355 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/%.c
1356 $(COMPILE.c) -o $$@ $<
1357 $(CTFCONVERT_O)

1359 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/ipgpc/%.c
1360 $(COMPILE.c) -o $$@ $<
1361 $(CTFCONVERT_O)

1363 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/dlcosmk/%.c
1364 $(COMPILE.c) -o $$@ $<
1365 $(CTFCONVERT_O)

1367 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/flowacct/%.c
1368 $(COMPILE.c) -o $$@ $<
1369 $(CTFCONVERT_O)

1371 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/dscpmk/%.c
1372 $(COMPILE.c) -o $$@ $<
1373 $(CTFCONVERT_O)

1375 $(OBJSDIR)/%.o: $(UTSBASE)/common/ipp/meters/%.c
1376 $(COMPILE.c) -o $$@ $<
1377 $(CTFCONVERT_O)

1379 $(OBJSDIR)/%.o: $(UTSBASE)/common/kiconv/kiconv_emea/%.c
1380 $(COMPILE.c) -o $$@ $<
1381 $(CTFCONVERT_O)

```

```

1383 $(OBJSDIR)/%.o: $(UTSBASE)/common/kiconv/kiconv_ja/%.c
1384 $(COMPILE.c) -o $$@ $<
1385 $(CTFCONVERT_O)

1387 $(OBJSDIR)/%.o: $(UTSBASE)/common/kiconv/kiconv_ko/%.c
1388 $(COMPILE.c) -o $$@ $<
1389 $(CTFCONVERT_O)

1391 $(OBJSDIR)/%.o: $(UTSBASE)/common/kiconv/kiconv_sc/%.c
1392 $(COMPILE.c) -o $$@ $<
1393 $(CTFCONVERT_O)

1395 $(OBJSDIR)/%.o: $(UTSBASE)/common/kiconv/kiconv_tc/%.c
1396 $(COMPILE.c) -o $$@ $<
1397 $(CTFCONVERT_O)

1399 $(OBJSDIR)/%.o: $(UTSBASE)/common/kmdb/%.c
1400 $(COMPILE.c) -o $$@ $<
1401 $(CTFCONVERT_O)

1403 $(OBJSDIR)/%.o: $(UTSBASE)/common/ktli/%.c
1404 $(COMPILE.c) -o $$@ $<
1405 $(CTFCONVERT_O)

1407 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/scsi/adapters/iscsi/%.c
1408 $(COMPILE.c) -o $$@ $<
1409 $(CTFCONVERT_O)

1411 $(OBJSDIR)/%.o: $(COMMONBASE)/iscsi/%.c
1412 $(COMPILE.c) -o $$@ $<
1413 $(CTFCONVERT_O)

1415 $(OBJSDIR)/%.o: $(UTSBASE)/common/inet/kifconf/%.c
1416 $(COMPILE.c) -o $$@ $<
1417 $(CTFCONVERT_O)

1419 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/vr/%.c
1420 $(COMPILE.c) -o $$@ $<
1421 $(CTFCONVERT_O)

1423 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/yge/%.c
1424 $(COMPILE.c) -o $$@ $<
1425 $(CTFCONVERT_O)

1427 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/virtio/%.c
1428 $(COMPILE.c) -o $$@ $<
1429 $(CTFCONVERT_O)

1431 $(OBJSDIR)/%.o: $(UTSBASE)/common/io/vioblk/%.c
1432 $(COMPILE.c) -o $$@ $<
1433 $(CTFCONVERT_O)

1435 #
1436 # krtld must refer to its own bzero/bcopy until the kernel is fully linked
1437 #
1438 $(OBJSDIR)/bootrd.o := CPPFLAGS += -DKOBJ_OVERRIDES
1439 $(OBJSDIR)/doreloc.o := CPPFLAGS += -DKOBJ_OVERRIDES
1440 $(OBJSDIR)/kobj.o := CPPFLAGS += -DKOBJ_OVERRIDES
1441 $(OBJSDIR)/kobj_boot.o := CPPFLAGS += -DKOBJ_OVERRIDES
1442 $(OBJSDIR)/kobj_bootflags.o := CPPFLAGS += -DKOBJ_OVERRIDES
1443 $(OBJSDIR)/kobj_convrelstr.o := CPPFLAGS += -DKOBJ_OVERRIDES
1444 $(OBJSDIR)/kobj_isa.o := CPPFLAGS += -DKOBJ_OVERRIDES
1445 $(OBJSDIR)/kobj_kdi.o := CPPFLAGS += -DKOBJ_OVERRIDES
1446 $(OBJSDIR)/kobj_lm.o := CPPFLAGS += -DKOBJ_OVERRIDES
1447 $(OBJSDIR)/kobj_reloc.o := CPPFLAGS += -DKOBJ_OVERRIDES

```

```

1448 $(OBJSDIR)/kobj_stubs.o      := CPPFLAGS += -DKOBJ_OVERRIDES
1449 $(OBJSDIR)/kobj_subr.o       := CPPFLAGS += -DKOBJ_OVERRIDES

1451 $(OBJSDIR)/%.o:              $(UTSBASE)/common/krtld/%.c
1452     $(COMPILE.c) -o $@ $<
1453     $(CTFCONVERT_O)

1455 $(OBJSDIR)/%.o:              $(COMMONBASE)/list/%.c
1456     $(COMPILE.c) -o $@ $<
1457     $(CTFCONVERT_O)

1459 $(OBJSDIR)/%.o:              $(COMMONBASE)/lvm/%.c
1460     $(COMPILE.c) -o $@ $<
1461     $(CTFCONVERT_O)

1463 $(OBJSDIR)/%.o:              $(COMMONBASE)/lzma/%.c
1464     $(COMPILE.c) -o $@ $<
1465     $(CTFCONVERT_O)

1467 $(OBJSDIR)/%.o:              $(COMMONBASE)/crypto/md4/%.c
1468     $(COMPILE.c) -o $@ $<
1469     $(CTFCONVERT_O)

1471 $(OBJSDIR)/%.o:              $(COMMONBASE)/crypto/md5/%.c
1472     $(COMPILE.c) -o $@ $<
1473     $(CTFCONVERT_O)

1475 $(OBJSDIR)/%.o:              $(COMMONBASE)/net/dhbj/%.c
1476     $(COMPILE.c) -o $@ $<
1477     $(CTFCONVERT_O)

1479 $(OBJSDIR)/%.o:              $(COMMONBASE)/nvpair/%.c
1480     $(COMPILE.c) -o $@ $<
1481     $(CTFCONVERT_O)

1483 $(OBJSDIR)/%.o:              $(UTSBASE)/common/os/%.c
1484     $(COMPILE.c) -o $@ $<
1485     $(CTFCONVERT_O)

1487 $(OBJSDIR)/%.o:              $(UTSBASE)/common/pcmcia/cis/%.c
1488     $(COMPILE.c) -o $@ $<
1489     $(CTFCONVERT_O)

1491 $(OBJSDIR)/%.o:              $(UTSBASE)/common/pcmcia/cs/%.c
1492     $(COMPILE.c) -o $@ $<
1493     $(CTFCONVERT_O)

1495 $(OBJSDIR)/%.o:              $(UTSBASE)/common/pcmcia/nexus/%.c
1496     $(COMPILE.c) -o $@ $<
1497     $(CTFCONVERT_O)

1499 $(OBJSDIR)/%.o:              $(UTSBASE)/common/pcmcia/pcs/%.c
1500     $(COMPILE.c) -o $@ $<
1501     $(CTFCONVERT_O)

1503 $(OBJSDIR)/%.o:              $(UTSBASE)/common/rpc/%.c
1504     $(COMPILE.c) -o $@ $<
1505     $(CTFCONVERT_O)

1507 $(OBJSDIR)/%.o:              $(UTSBASE)/common/rpc/sec/%.c
1508     $(COMPILE.c) -o $@ $<
1509     $(CTFCONVERT_O)

1511 $(OBJSDIR)/%.o:              $(UTSBASE)/common/rpc/sec_gss/%.c
1512     $(COMPILE.c) -o $@ $<
1513     $(CTFCONVERT_O)

```

```

1515 $(OBJSDIR)/%.o:              $(COMMONBASE)/crypto/shal/%.c
1516     $(COMPILE.c) -o $@ $<
1517     $(CTFCONVERT_O)

1519 $(OBJSDIR)/%.o:              $(COMMONBASE)/crypto/sha2/%.c
1520     $(COMPILE.c) -o $@ $<
1521     $(CTFCONVERT_O)

1523 $(OBJSDIR)/%.o:              $(UTSBASE)/common/syscall/%.c
1524     $(COMPILE.c) -o $@ $<
1525     $(CTFCONVERT_O)

1527 $(OBJSDIR)/%.o:              $(UTSBASE)/common/tnf/%.c
1528     $(COMPILE.c) -o $@ $<
1529     $(CTFCONVERT_O)

1531 $(OBJSDIR)/%.o:              $(COMMONBASE)/tsol/%.c
1532     $(COMPILE.c) -o $@ $<
1533     $(CTFCONVERT_O)

1535 $(OBJSDIR)/%.o:              $(COMMONBASE)/util/%.c
1536     $(COMPILE.c) -o $@ $<
1537     $(CTFCONVERT_O)

1539 $(OBJSDIR)/%.o:              $(COMMONBASE)/unicode/%.c
1540     $(COMPILE.c) -o $@ $<
1541     $(CTFCONVERT_O)

1543 $(OBJSDIR)/%.o:              $(UTSBASE)/common/vm/%.c
1544     $(COMPILE.c) -o $@ $<
1545     $(CTFCONVERT_O)

1547 $(OBJSDIR)/%.o:              $(UTSBASE)/common/zmod/%.c
1548     $(COMPILE.c) -o $@ $<
1549     $(CTFCONVERT_O)

1551 $(OBJSDIR)/zlib_obj.o:        $(ZLIB_OBJS:=$(OBJSDIR)/%)
1552     $(LD) -r -Breduce -M$(UTSBASE)/common/zmod/mapfile -o $@ \
1553         $(ZLIB_OBJS:=$(OBJSDIR)/%)
1554     $(CTFMERGE) -t -f -L VERSION -o $@ $(ZLIB_OBJS:=$(OBJSDIR)/%)

1556 $(OBJSDIR)/%.o:              $(UTSBASE)/common/io/hxge/%.c
1557     $(COMPILE.c) -o $@ $<
1558     $(CTFCONVERT_O)

1560 $(OBJSDIR)/%.o:              $(UTSBASE)/common/io/tpm/%.c
1561     $(COMPILE.c) -o $@ $<
1562     $(CTFCONVERT_O)

1564 $(OBJSDIR)/%.o:              $(UTSBASE)/common/io/tpm/%.s
1565     $(COMPILE.s) -o $@ $<

1567 $(OBJSDIR)/bz2%.o:           $(COMMONBASE)/bzip2/%.c
1568     $(COMPILE.c) -o $@ -I$(COMMONBASE)/bzip2 $<
1569     $(CTFCONVERT_O)

1571 BZ2LINT = -erroff=%all -I$(UTSBASE)/common/bzip2

1573 $(LINTSDIR)/bz2%.ln:          $(COMMONBASE)/bzip2/%.c
1574     @$(LHEAD) $(LINT.c) -C $(LINTSDIR)/`basename $@ .ln` $(BZ2LINT) $< $(

1576 #
1577 # SVM
1578 #

```





```

1712      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1714 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/disp/%.c
1715      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1717 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/dtrace/%.c
1718      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1720 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/exacct/%.c
1721      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1723 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/exec/aout/%.c
1724      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1726 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/exec/elf/%.c
1727      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1729 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/exec/intp/%.c
1730      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1732 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/exec/shbin/%.c
1733      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1735 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/exec/java/%.c
1736      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1738 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/%.c
1739      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1741 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/autofs/%.c
1742      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1744 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/cacheufs/%.c
1745      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1747 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/ctfs/%.c
1748      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1750 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/doorfs/%.c
1751      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1753 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/dcfvfs/%.c
1754      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1756 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/devfs/%.c
1757      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1759 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/dev/%.c
1760      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1762 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/fd/%.c
1763      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1765 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/fifofs/%.c
1766      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1768 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/hsfs/%.c
1769      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1771 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/lofs/%.c
1772      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1774 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/mntfs/%.c
1775      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1777 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/namefs/%.c

```

```

1778      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1780 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/smbsrv/%.c
1781      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1783 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/smbsrv/%.c
1784      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1786 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/nfs/%.c
1787      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1789 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/objfs/%.c
1790      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1792 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/pcfs/%.c
1793      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1795 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/portfs/%.c
1796      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1798 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/proc/%.c
1799      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1801 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/sharefs/%.c
1802      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1804 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/smbclnt/%.c
1805      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1807 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/smbclnt/net smb/%.c
1808      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1810 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/smbclnt/smbfs/%.c
1811      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1813 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/sockfs/%.c
1814      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1816 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/specfs/%.c
1817      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1819 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/swapfs/%.c
1820      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1822 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/tmpfs/%.c
1823      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1825 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/udfs/%.c
1826      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1828 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/ufs/%.c
1829      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1831 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/ufs_log/%.c
1832      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1834 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/vscan/%.c
1835      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1837 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/zfs/%.c
1838      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1840 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/fs/zut/%.c
1841      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1843 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/xattr/%.c

```

```

1844      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1846 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/zfs/%.c
1847      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1849 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/gssapi/%.c
1850      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1852 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/gssapi/mechs/dummy/%.c
1853      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1855 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/%.c
1856      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1858 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/%.c
1859      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1861 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/des/%.c
1862      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1864 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/dk/%.c
1865      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1867 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/os/%.c
1868      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1870 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/arcfour/%.c
1871      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1873 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/enc_provider/%.c
1874      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1876 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/hash_provider/%.c
1877      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1879 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/keyhash_provider/%.c
1880      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1882 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/raw/%.c
1883      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1885 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/crypto/old/%.c
1886      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1888 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/krb5/krb/%.c
1889      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1891 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/krb5/os/%.c
1892      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1894 $(LINTS_DIR)/%.ln:      $(KMECHKRB5_BASE)/mech/%.c
1895      @$(LHEAD) $(LINT.c) $(KGSSDFLAGS) $< $(LTAIL))
1897 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/idmap/%.c
1898      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1900 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/%.c
1901      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1903 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/sockmods/%.c
1904      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1906 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/arp/%.c
1907      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1909 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/ip/%.c

```

```

1910      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1912 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/ipnet/%.c
1913      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1915 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/iptun/%.c
1916      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1918 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/ipf/%.c
1919      @$(LHEAD) $(LINT.c) $(IPFFLAGS) $< $(LTAIL))
1921 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/kssl/%.c
1922      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1924 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/net/patricia/%.c
1925      @$(LHEAD) $(LINT.c) $(IPFFLAGS) $< $(LTAIL))
1927 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/udp/%.c
1928      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1930 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/sctp/%.c
1931      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1933 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/tcp/%.c
1934      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1936 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/ilb/%.c
1937      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1939 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/nca/%.c
1940      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1942 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/dlpistub/%.c
1943      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1945 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/%.c
1946      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1948 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/1394/%.c
1949      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1951 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/1394/adapters/%.c
1952      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1954 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/1394/targets/av1394/%.c
1955      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1957 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/1394/targets/dcam1394/%.c
1958      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1960 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/1394/targets/scsal394/%.c
1961      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1963 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sbp2/%.c
1964      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1966 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/aac/%.c
1967      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1969 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/afe/%.c
1970      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1972 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/atge/%.c
1973      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1975 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/arn/%.c

```

```

1976      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1978 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ath/%.c
1979      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1981 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/atu/%.c
1982      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1984 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/impl/%.c
1985      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1987 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/ac97/%.c
1988      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1990 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audio1575/%.c
1991      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1993 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audio810/%.c
1994      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1996 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiocmi/%.c
1997      @$(LHEAD) $(LINT.c) $< $(LTAIL))
1999 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiocmihd/%.c
2000      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2002 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audioens/%.c
2003      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2005 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audioemu10k/%.c
2006      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2008 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiohd/%.c
2009      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2011 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audioixp/%.c
2012      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2014 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiols/%.c
2015      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2017 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiopci/%.c
2018      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2020 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiopl6x/%.c
2021      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2023 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiosolo/%.c
2024      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2026 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiots/%.c
2027      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2029 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiovia823x/%.c
2030      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2032 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/audio/drv/audiovia97/%.c
2033      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2035 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/bfe/%.c
2036      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2038 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/bpf/%.c
2039      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2041 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/bge/%.c

```

```

2042      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2044 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/blkdev/%.c
2045      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2047 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cardbus/%.c
2048      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2050 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/lu/stmf_sbd/%.c
2051      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2053 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/fct/%.c
2054      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2056 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/qlt/%.c
2057      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2059 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/srpt/%.c
2060      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2062 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/iscsit/%.c
2063      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2065 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/fcoet/%.c
2066      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2068 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/iscsit/%.c
2069      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2071 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/port/pppt/%.c
2072      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2074 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/comstar/stmf/%.c
2075      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2077 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/dld/%.c
2078      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2080 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/dls/%.c
2081      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2083 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/dmfe/%.c
2084      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2086 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/drm/%.c
2087      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2089 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/efe/%.c
2090      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2092 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/elxl/%.c
2093      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2095 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fcoe/%.c
2096      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2098 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fsd/%.c
2099      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2101 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/hme/%.c
2102      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2104 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pciex/%.c
2105      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2107 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/hotplug/hpcsvc/%.c

```

```

2108      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2110 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pciex/hotplug/%.c
2111      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2113 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/hotplug/pcihp/%.c
2114      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2116 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/rds/%.c
2117      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2119 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/rdsv3/%.c
2120      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2122 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/iser/%.c
2123      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2125 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/ibd/%.c
2126      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2128 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/eoib/%.c
2129      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2131 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/of/sol_ofs/%.c
2132      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2134 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/of/sol_ucma/%.c
2135      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2137 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/of/sol_umad/%.c
2138      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2140 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/of/sol_uverbs/%.
2141      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2143 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/sdp/%.c
2144      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2146 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/mgt/ibcm/%.c
2147      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2149 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/mgt/ibdm/%.c
2150      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2152 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/mgt/ibdma/%.c
2153      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2155 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/mgt/ibmf/%.c
2156      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2158 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/ibnex/%.c
2159      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2161 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/ibt1/%.c
2162      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2164 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/adapters/tavor/%.c
2165      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2167 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/adapters/hermon/%.c
2168      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2170 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ib/clients/daplt/%.c
2171      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2173 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/iscsi/%.c

```

```

2174      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2176 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/idm/%.c
2177      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2179 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ipw/%.c
2180      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2182 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/iwh/%.c
2183      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2185 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/iwi/%.c
2186      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2188 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/iwk/%.c
2189      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2191 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/iwp/%.c
2192      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2194 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/kb8042/%.c
2195      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2197 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/kbtrans/%.c
2198      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2200 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ksocket/%.c
2201      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2203 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/aggr/%.c
2204      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2206 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lp/%.c
2207      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2209 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/hotspares/%.c
2210      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2212 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/md/%.c
2213      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2215 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/mirror/%.c
2216      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2218 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/raid/%.c
2219      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2221 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/softpart/%.c
2222      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2224 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/stripes/%.c
2225      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2227 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/notify/%.c
2228      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2230 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/lvm/trans/%.c
2231      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2233 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mac/%.c
2234      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2236 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mac/plugins/%.c
2237      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2239 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mega_sas/%.c

```

```

2240      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2242 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mii/%.c
2243   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2245 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mr_sas/%.c
2246   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2248 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/mpt_sas/%.c
2249   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2251 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mxfe/%.c
2252   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2254 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mwl/%.c
2255   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2257 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/mwl/mwl_fw/%.c
2258   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2260 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/net80211/%.c
2261   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2263 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/nge/%.c
2264   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2266 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/nxge/%.c
2267   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2269 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/nxge/%.s
2270   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2272 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/nxge/npi/%.c
2273   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2275 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pci-ide/%.c
2276   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2278 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pcmcia/%.c
2279   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2281 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pcan/%.c
2282   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2284 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pcn/%.c
2285   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2287 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/pcwl/%.c
2288   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2290 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ppp/sppp/%.c
2291   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2293 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ppp/spppasyn/%.c
2294   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2296 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ppp/sppptun/%.c
2297   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2299 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ral/%.c
2300   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2302 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rge/%.c
2303   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2305 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rtls/%.c

```

```

2306      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2308 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rsm/%.c
2309   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2311 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rtw/%.c
2312   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2314 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rum/%.c
2315   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2317 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rwd/%.c
2318   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2320 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/rwn/%.c
2321   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2323 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sata/adapters/ahci/%.c
2324   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2326 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sata/adapters/nv_sata/%.c
2327   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2329 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sata/adapters/si3124/%.c
2330   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2332 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sata/impl/%.c
2333   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2335 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/%.c
2336   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2338 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/blk2scsa/%.c
2339   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2341 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/pmcs/%.c
2342   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2344 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/scsi_vhci/%.c
2345   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2347 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/scsi_vhci/fop
2348   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2350 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/ulp/%.c
2351   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2353 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/impl/%.c
2354   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2356 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/fca/qlc/%.c
2357   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2359 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/fca/qlge/%.c
2360   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2362 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/fca/emlxs/%.c
2363   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2365 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/fca/oce/%.c
2366   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2368 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/fibre-channel/fca/fcoei/%.c
2369   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2371 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/conf/%.c

```

```

2372      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2374 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/impl/%.c
2375      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2377 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/targets/%.c
2378      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2380 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sdcard/adapters/sdhost/%.c
2381      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2383 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sdcard/impl/%.c
2384      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2386 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sdcard/targets/sdcard/%.c
2387      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2389 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/sfe/%.c
2390      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2392 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/simnet/%.c
2393      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2395 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/softmac/%.c
2396      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2398 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/uath/%.c
2399      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2401 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/uath/uath_fw/%.c
2402      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2404 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ural/%.c
2405      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2407 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/urtw/%.c
2408      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2410 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/audio/usb_ac/%.
2411      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2413 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/audio/usb_as/%.
2414      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2416 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/audio/usb_ah/%.
2417      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2419 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbskel/%.c
2420      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2422 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/video/usbvc/%.c
2423      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2425 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/hwarc/%.c
2426      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2428 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/hid/%.c
2429      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2431 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/hidparser/%.c
2432      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2434 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/uskbkm/%.c
2435      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2437 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbms/%.c

```

```

2438      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2440 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbinput/usbwcm
2441      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2443 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/ugen/%.c
2444      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2446 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/printer/%.c
2447      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2449 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbser/%.c
2450      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2452 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbser/usbsacm/
2453      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2455 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbser/usbftdi/
2456      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2458 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbser/usbser_k
2459      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2461 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbser/usbsprl/
2462      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2464 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/wusb_df/%.c
2465      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2467 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/hwal480_fw/%.c
2468      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2470 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/wusb_ca/%.c
2471      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2473 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/clients/usbecm/%.c
2474      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2476 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/hcd/openhci/%.c
2477      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2479 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/hcd/ehci/%.c
2480      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2482 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/hcd/uhci/%.c
2483      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2485 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/hubd/%.c
2486      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2488 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/scsa2usb/%.c
2489      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2491 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/usb_mid/%.c
2492      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2494 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/usb_ia/%.c
2495      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2497 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/usba/%.c
2498      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2500 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/usba10/%.c
2501      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2503 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/uwb/uwba/%.c

```

```

2504      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2506 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/usb/hwa/hwahc/%.c
2507   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2509 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/vuidmice/%.c
2510   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2512 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/vnic/%.c
2513   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2515 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/wpi/%.c
2516   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2518 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/zyd/%.c
2519   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2521 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/chxge/com/%.c
2522   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2524 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/chxge/%.c
2525   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2527 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cxgbe/common/%.c
2528   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2530 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cxgbe/shared/%.c
2531   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2533 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cxgbe/firmware/%.c
2534   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2536 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cxgbe/t4nex/%.c
2537   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2539 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/cxgbe/cxgbe/%.c
2540   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2542 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ixgb/%.c
2543   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2545 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/xge/drv/%.c
2546   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2548 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/xge/hal/xgehal/%.c
2549   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2551 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/e1000g/%.c
2552   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2554 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/igb/%.c
2555   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2557 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/iprb/%.c
2558   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2560 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ixgbe/%.c
2561   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2563 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/ntxn/%.c
2564   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2566 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/myril0ge/drv/%.c
2567   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2569 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/%.c

```

```

2570      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2572 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/ipgpc/%.c
2573   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2575 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/dlcosmk/%.c
2576   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2578 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/flowacct/%.c
2579   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2581 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/dscpmk/%.c
2582   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2584 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ipp/meters/%.c
2585   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2587 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kiconv/kiconv_emea/%.c
2588   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2590 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kiconv/kiconv_ja/%.c
2591   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2593 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kiconv/kiconv_ko/%.c
2594   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2596 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kiconv/kiconv_sc/%.c
2597   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2599 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kiconv/kiconv_tc/%.c
2600   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2602 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/kmdb/%.c
2603   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2605 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/krtld/%.c
2606   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2608 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/ktli/%.c
2609   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2611 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/list/%.c
2612   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2614 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/lvm/%.c
2615   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2617 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/lzma/%.c
2618   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2620 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/crypto/md4/%.c
2621   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2623 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/crypto/md5/%.c
2624   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2626 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/net/dhcp/%.c
2627   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2629 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/nvpair/%.c
2630   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2632 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/os/%.c
2633   @$(LHEAD) $(LINT.c) $< $(LTAIL))
2635 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/rpc/%.c

```

```

2636      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2638 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/pcmcia/cs/%.c
2639      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2641 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/pcmcia/cis/%.c
2642      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2644 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/pcmcia/nexus/%.c
2645      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2647 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/pcmcia/pcs/%.c
2648      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2650 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/rpc/%.c
2651      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2653 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/rpc/sec/%.c
2654      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2656 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/rpc/sec_gss/%.c
2657      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2659 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/crypto/shal/%.c
2660      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2662 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/crypto/sha2/%.c
2663      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2665 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/syscall/%.c
2666      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2668 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/tnf/%.c
2669      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2671 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/tsol/%.c
2672      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2674 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/util/%.c
2675      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2677 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/unicode/%.c
2678      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2680 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/vm/%.c
2681      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2683 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/scsi/adapters/iscsi/%.c
2684      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2686 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/iscsi/%.c
2687      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2689 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/inet/kifconf/%.c
2690      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2692 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/virtio/%.c
2693      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2695 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/vioblk/%.c
2696      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2698 ZMODLINTFLAGS = -erroff=E_CONSTANT_CONDITION
2700 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/zmod/%.c
2701      @$(LHEAD) $(LINT.c) $(ZMODLINTFLAGS) $< $(LTAIL))

```

```

2703 $(LINTS_DIR)/zlib_obj.ln:      $(ZLIB_OBJS:%.o=$(LINTS_DIR)/%.ln) \
2704      $(UTSBASE)/common/zmod/zlib_lint.c
2705      @$(LHEAD) $(LINT.c) -C $(LINTS_DIR)/zlib_obj \
2706      $(UTSBASE)/common/zmod/zlib_lint.c $(LTAIL))
2708 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/hxge/%.c
2709      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2711 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/tpm/%.c
2712      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2714 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/tpm/%.s
2715      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2717 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/vr/%.c
2718      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2720 $(LINTS_DIR)/%.ln:      $(UTSBASE)/common/io/yge/%.c
2721      @$(LHEAD) $(LINT.c) $< $(LTAIL))
2723 $(LINTS_DIR)/%.ln:      $(COMMONBASE)/fsreparse/%.c
2724      @$(LHEAD) $(LINT.c) $< $(LTAIL))

```



```

*****
32997 Tue Sep 3 18:09:41 2013
new/usr/src/uts/common/fs/fsh.c
fsh, fsd, libfsd, fsdam
*****
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 Damian Bogel. All rights reserved.
14 */

16 #include <sys/debug.h>
17 #include <sys/errno.h>
18 #include <sys/fsh.h>
19 #include <sys/fsh_impl.h>
20 #include <sys/id_space.h>
21 #include <sys/kmem.h>
22 #include <sys/ksynch.h>
23 #include <sys/list.h>
24 #include <sys/sunddi.h>
25 #include <sys/sysmacros.h>
26 #include <sys/types.h>
27 #include <sys/vfs.h>
28 #include <sys/vnode.h>

30 /*
31  * Filesystem hook framework (fsh)
32  *
33  * 1. Abstract.
34  * The main goal of the filesystem hook framework is to provide an easy way to
35  * inject client-defined behaviour into vfs/vnode calls. fsh works on
36  * vfs_t granularity.
37  *
38  *
39  * 2. Overview.
40  * fsh_t is the main object in the fsh. An fsh_t is a structure containing:
41  * - pointers to hooking functions (named after corresponding
42  *   vnodeops/vfsops)
43  * - a pointer to an argument to pass (this is shared for all the
44  *   hooks in a given fsh_t)
45  * - a pointer to the *hook remove callback* - it's being fired after a
46  *   hook is removed and the hook has stopped executing. It's safe to destroy
47  *   any data associated with this hook.
48  *
49  * The information from fsh_t is copied by the fsh and an fsh_handle_t
50  * is returned. It should be used for further removing.
51  *
52  *
53  * 3. Usage.
54  * It is expected that vfs_t/vnode_t that are passed to fsh_foo() functions
55  * are held by the caller when needed. fsh does no vfs_t/vnode_t locking.
56  *
57  * fsh_t is a structure filled out by the client. If a client does not want
58  * to add/remove a hook for function foo(), he should fill the foo field of
59  * fsh_t with NULL. Every hook has a type of corresponding vfsop/vnodeop with
60  * two additional arguments:
61  * - fsh_int_t *fsh_int - this argument MUST be passed to

```

```

62  *   hook_next_foo(). fsh wouldn't know which hook to execute next
63  *   without it
64  *   - void *arg - this is the argument passed with fsh_t during
65  *   installation
66  *   - void (*remove_cb)(void *, fsh_handle_t) - hook remove callback
67  *   (mentioned earlier); it's first argument is arg, the second is the
68  *   handle
69  *
70  * After installation, an fsh_handle_t is returned to the caller.
71  *
72  * Every hook function is responsible for passing the control to the next
73  * hook associated with a particular call. In order to provide an easy way to
74  * modify the behaviour of a function call both before and after the
75  * underlying vfsop/vnodeop (or next hook) execution, a hook has to call
76  * fsh_next_foo() at some point. This function does necessary internal
77  * operations and calls the next hook, until there's no hook left, then it
78  * calls the underlying vfsop/vnodeop.
79  * Example:
80  * my_freefs(fsh_int_t *fsh_int, void *arg, vfs_t *vfsp) {
81  *     cmn_err(CE_NOTE, "freefs called!\n");
82  *     return (fsh_next_freefs(fsh_int, vfsp));
83  * }
84  *
85  *
86  * A client might want to fire callbacks when vfs_t's are being mounted
87  * or freed. There's an fsh_callback_t structure provided to install such
88  * callbacks along with the API.
89  * It is legal to call fsh_hook_{install,remove}() inside a mount callback
90  * WITHOUT holding the vfs_t.
91  *
92  * After vfs_t's free callback returns, all the handles associated with the
93  * hooks installed on this vfs_t are invalid and must not be used.
94  *
95  *
96  * 4. API
97  * None of the APIs should be called during interrupt context above lock
98  * level. The only exceptions are fsh_next_foo() functions, which do not use
99  * locks.
100 *
101 * a) fsh.h
102 * Any of these functions could be called inside a hook or a hook remove
103 * callback.
104 * fsh_callback_{install,remove}() must not be called inside a {mount,free}
105 * callback. Doing so will cause a deadlock. Other functions can be called
106 * inside {mount,free} callbacks.
107 *
108 * fsh_fs_enable(vfs_t *vfsp)
109 * fsh_fs_disable(vfs_t *vfsp)
110 *     Enables/disables fsh for a given vfs_t.
111 *
112 * fsh_hook_install(vfs_t *vfsp, fsh_t *hooks)
113 *     Installs hooks on vfsp filesystem.
114 *     It's important that hooks are executed in LIFO installation order,
115 *     which means that if there are hooks A and B installed in this order, B
116 *     is going to be executed before A.
117 *     It returns a correct handle, or (-1) if hook/callback limit exceeded.
118 *     The handle is valid until a free callback returns or an explicit call
119 *     to fsh_hook_remove().
120 *
121 * fsh_hook_remove(fsh_handle_t handle)
122 *     Removes a hook and invalidates the handle.
123 *     It is guaranteed that after this function returns, calls to
124 *     vnodeops/vfsops won't go through this hook, although there might be
125 *     some threads still executing this hook. When hook remove callback is
126 *     fired, it is guaranteed that the hook won't be executed anymore. It is
127 *     safe to remove all the internal data associated with this hook inside

```

```

128 *   the hook remove callback. The hook remove callback could be called
129 *   inside fsh_hook_remove().
130 *
131 * fsh_next_foo(fsh_int_t *fsh_int, void *arg, ARGUMENTS)
132 * This is the function which should be called once in every hook. It
133 * does the necessary internal operations and passes control to the
134 * next hook or, if there's no hook left, to the underlying
135 * vfsop/vnodeop.
136 *
137 * fsh_callback_install(fsh_callback_t *callback)
138 * fsh_callback_remove(fsh_callback_handle_t handle)
139 * Installs/removes callbacks for vfs_t mount/free. The mount callback
140 * is executed right before domount() returns. The free callback is
141 * called right before VFS_FREEVFS() is called.
142 * The fsh_callback_install() returns a correct handle, or (-1) if
143 * hook/callback limit exceeded.
144 *
145 * b) fsh_impl.h (for vfs.c and vnode.c only)
146 * fsh_init()
147 * This call has to be done in vfsinit(). It initialises the fsh. It
148 * is absolutely necessary that this call is made before any other fsh
149 * operation.
150 *
151 * fsh_exec_mount_callbacks(vfs_t *vfsp)
152 * fsh_exec_free_callbacks(vfs_t *vfsp)
153 * Used to execute all fsh callbacks for {mount,free} of a vfs_t.
154 *
155 * fsh_fsrec_destroy(struct fsh_fsrecord *fsrecp)
156 * Destroys an fsh_fsrecord structure. All the hooks installed on this
157 * vfs_t are then destroyed. free callback is called before this function.
158 *
159 * fsh_foo(ARGUMENTS)
160 * Function used to start executing the hook chain for a given call.
161 *
162 *
163 * 5. Internals.
164 * fsh_int_t is an internal hook structure. It is reference counted.
165 * fshi_hold() and fshi_rele() should be used whenever needed.
166 * fsh_int_t entries are elements of both fsh_map (global) and fshfsr_list
167 * (local to vfs_t). All entries are unique and are identified by fshi_handle.
168 *
169 * fsh_int_t properties:
170 * - fsh_hook_install() sets the ref. counter to 1 and adds it to both
171 * fsh_map and fshfsr_list
172 * - fsh_hook_remove() decreases the ref. counter by 1, removes the hook
173 * from fsh_map and marks the hook as *doomed*
174 * - if fsh_int_t is on the fshfsr_list, it's alive and there is a thread
175 * executing it
176 * - if fsh_int_t is marked as *doomed*, the reference counter is not
177 * be increased and thus no thread can acquire this fsh_int_t
178 * - ref. counter can drop to 0 only after an fsh_hook_remove() call; this
179 * also means that the fsh_int_t is *doomed* and isn't a part of fsh_map
180 * - fsh_int_t could be also destroyed without fsh_hook_remove() call,
181 * that happens only inside fsh_fsrec_destroy() where it is guaranteed
182 * that there is no thread executing the hook
183 *
184 *
185 * fsh_fsrecord_t is a structure which lives inside a vfs_t.
186 * fsh_fsrecord_t contains:
187 * - an rw-lock that protects the structure
188 * - a list of hooks installed on this vfs_t
189 * - a flag which tells whether fsh is enabled on this vfs_t
190 *
191 *
192 * fsh_prepare_fsrec rule:
193 * Every function that needs vfsp->vfs_fshrecord has to call

```

```

194 * fsh_prepare_fsrec() first. If and only if the call is made, it is safe to
195 * use vfsp->vfs_fshrecord.
196 *
197 * Unfortunately, because of unexpected behaviour of some filesystems (no use
198 * of vfs_alloc()/vfs_init()) there's no good place to initialise the
199 * fsh_fshrecord_t structure. The approach being used here is to check if it's
200 * initialised in every call. Because of the fact that no lock could be used
201 * here (the same problem with initialisation), a spinlock is used. This is
202 * explained in more detail in a comment before fsh_prepare_fsrec(). After
203 * calling fsh_preapre_fsrec() it's completely safe to keep the vfs_fshrecord
204 * pointer locally, because it won't be changed until vfs_free() is called.
205 *
206 * The only exception from the fsh_prepare_fsrec() rule is vfs_free(),
207 * where there is expected that no other fsh calls would be made for the
208 * vfs_t that's being freed. That's why vfs_fshrecord could be only NULL or a
209 * valid pointer and could not be concurrently accessed.
210 *
211 * When there are no fsh functions (that use a particular fsh_fsrecord_t)
212 * executing, the vfs_fshrecord pointer won't be equal to fsh_res_ptr. It
213 * would be NULL or a pointer to an initialised fsh_fsrecord_t.
214 *
215 *
216 * Callbacks:
217 * Mount callbacks are executed by a call to fsh_exec_mount_callbacks() right
218 * before returning from domount()@vfs.c.
219 *
220 * Free callbacks are executed by a call to fsh_exec_free_callbacks() right
221 * before calling VFS_FREEVFS(), after vfs_t's reference count drops to 0.
222 *
223 *
224 * fsh_next_foo(fsh_int_t *fshi, ARGUMENTS)
225 * This function is quite simple. It takes the fsh_int_t and passes control
226 * to the next hook or to the underlying vnodeop/vfsop.
227 *
228 *
229 * 6. Locking
230 * a) public
231 * fsh does no vfs_t nor vnode_t locking. It is expected that whenever it is
232 * needed, the client does that.
233 *
234 * fsh_callback_{install,remove} must not be called inside a callback, because
235 * it will cause a deadlock.
236 *
237 * b) internal
238 * Locking diagram:
239 *
240 *   fsh_hook_install()   fsh_hook_remove()   fsh_fsrec_destroy()
241 *   |                   |                   |
242 *   +-----+           +-----+           +-----+
243 *   |                   |                   |
244 *   |                   |                   |
245 *   |                   |                   |
246 *   |                   |                   |
247 *   |                   |                   |
248 *   |                   |                   |
249 *   |                   |                   |
250 *   |                   |                   |
251 *   |                   |                   |
252 *   |                   |                   |
253 *   |                   |                   |
254 *   |                   |                   |
255 *   |                   |                   |
256 *   |                   |                   |
257 *   |                   |                   |
258 *   |                   |                   |
259 *   |                   |                   |

```

```

260 *      |
261 * fsh_read(),          fshi_rele()
262 * fsh_write(),
263 * ...,                Might be called from:
264 * fsh_next_read(),    fsh_hook_remove()
265 * fsh_next_write(),   fsh_read(), fsh_write(), ...
266 * ...,                fsh_next_read(), fsh_next_write(), ...
267 *
268 * fsh_lock is a global lock for administrative path (fsh_hook_install,
269 * fsh_hook_remove) and fsh_fsrec_destroy() (which is semi-administrative, since
270 * it destroys the unremoved hooks). It is used only when fsh_map needs to be
271 * locked. The usage of this lock guarantees that the data in fsh_map and
272 * fshfsr_lists is consistent.
273 */

276 /* Internals */
277 struct fsh_int {
278     fsh_handle_t    fshi_handle;
279     fsh_t           fshi_hooks;
280     vfs_t           *fshi_vfsp;

282     kmutex_t        fshi_lock;
283     uint64_t        fshi_ref;
284     uint64_t        fshi_doomed; /* changed inside fsh_lock */

286     /* next node in fshfsr_list */
287     list_node_t     fshi_next;

289     /* next node in fsh_map */
290     list_node_t     fshi_global;
291 };

293 typedef struct fsh_callback_int {
294     fsh_callback_t  fshci_cb;
295     fsh_callback_handle_t fshci_handle;
296     list_node_t     fshci_next;
297 } fsh_callback_int_t;

300 static kmutex_t fsh_lock;

302 /*
303 * fsh_fsrecord_t is the main internal structure. It's content is protected
304 * by fshfsr_lock. The fshfsr_list is a list of fsh_int_t hook entries for
305 * the vfs_t that contains the fsh_fsrecord_t.
306 */
307 struct fsh_fsrecord {
308     krwlock_t       fshfsr_lock;
309     int             fshfsr_enabled;
310     list_t          fshfsr_list;
311 };

313 /*
314 * Global list of fsh_int_t. Protected by fsh_lock.
315 */
316 static list_t fsh_map;

318 /*
319 * Global list of fsh_callback_int_t.
320 */
321 static krwlock_t fsh_cblist_lock;
322 static list_t fsh_cblist;

324 /*
325 * A reserved pointer for fsh purposes. It is used because of the method

```

```

326 * chosen for solving concurrency issues with vfs_fsrecord. The full
327 * explanation is in the big theory statement at the beginning of this
328 * file and above fsh_fsrec_prepare(). It is initialised in fsh_init().
329 */
330 static void *fsh_res_ptr;

332 static fsh_fsrecord_t *fsh_fsrec_create();

334 int fsh_limit = INT_MAX;
335 static id_space_t *fsh_idspace;

337 /*
338 * fsh_prepare_fsrec()
339 *
340 * Important note:
341 * Before using this function, fsh_init() MUST be called. We do that in
342 * vfsinit()@vfs.c.
343 *
344 * One would ask, why isn't the vfsp->vfs_fsrecord initialised when the
345 * vfs_t is created. Unfortunately, some filesystems (e.g. fifofs) do not
346 * call vfs_init() or even vfs_alloc(), It's possible that some unbundled
347 * filesystems could do the same thing. That's why this solution is
348 * introduced. It should be called before any code that needs access to
349 * vfs_fsrecord.
350 *
351 * Locking:
352 * There are no locks here, because there's no good place to initialise
353 * the lock. Concurrency issues are solved by using atomic instructions
354 * and a spinlock, which is spinning only once for a given vfs_t. Because
355 * of that, the usage of the spinlock isn't bad at all.
356 *
357 * How it works:
358 * a) if vfsp->vfs_fsrecord equals NULL, atomic_cas_ptr() changes it to
359 *    fsh_res_ptr. That's a signal for other threads, that the structure
360 *    is being initialised.
361 * b) if vfsp->vfs_fsrecord equals fsh_res_ptr, that means we have to wait,
362 *    because vfs_fsrecord is being initialised by another call.
363 * c) other cases:
364 *    vfs_fsrecord is already initialised, so we can use it. It won't change
365 *    until vfs_free() is called. It can't happen when someone is holding
366 *    the vfs_t, which is expected from the caller of fsh API.
367 */
368 static void
369 fsh_prepare_fsrec(vfs_t *vfsp)
370 {
371     fsh_fsrecord_t *fsrec;

373     while ((fsrec = atomic_cas_ptr(&vfsp->vfs_fsrecord, NULL,
374     fsh_res_ptr)) == fsh_res_ptr)
375         ;

377     if (fsrec == NULL)
378         atomic_swap_ptr(&vfsp->vfs_fsrecord, fsh_fsrec_create());
379 }

381 /*
382 * API for enabling/disabling fsh per vfs_t.
383 *
384 * A newly created vfs_t has fsh enabled by default. If one would want to change
385 * this behaviour, mount callbacks could be used.
386 *
387 * The caller is expected to hold the vfs_t.
388 *
389 * These functions must NOT be called in a hook.
390 */
391 void

```

```

392 fsh_fs_enable(vfs_t *vfsp)
393 {
394     fsh_prepare_fsrec(vfsp);

396     rw_enter(&vfsp->vfs_fshrecord->fshfsr_lock, RW_WRITER);
397     vfs->vfs_fshrecord->fshfsr_enabled = 1;
398     rw_exit(&vfsp->vfs_fshrecord->fshfsr_lock);
399 }

401 void
402 fsh_fs_disable(vfs_t *vfsp)
403 {
404     fsh_prepare_fsrec(vfsp);

406     rw_enter(&vfsp->vfs_fshrecord->fshfsr_lock, RW_WRITER);
407     vfs->vfs_fshrecord->fshfsr_enabled = 0;
408     rw_exit(&vfsp->vfs_fshrecord->fshfsr_lock);
409 }

411 /*
412  * API used for installing hooks. fsh_handle_t is returned for further
413  * actions (currently just removing) on this set of hooks.
414  *
415  * fsh_t fields:
416  * - arg - argument passed to every hook
417  * - remove_cb - remove callback, called after a hook is removed and all the
418  *   threads stops executing it
419  * - read, write, ... - pointers to hooks for corresponding vnodeops/vfsops;
420  *   if there is no hook desired for an operation, it should be set to
421  *   NULL
422  *
423  * It's important that the hooks are executed in LIFO installation order (they
424  * are added to the head of the hook list).
425  *
426  * The caller is expected to hold the vfs_t.
427  *
428  * Returns (-1) if hook/callback limit exceeded, handle otherwise.
429  */
430 fsh_handle_t
431 fsh_hook_install(vfs_t *vfsp, fsh_t *hooks)
432 {
433     fsh_handle_t    handle;
434     fsh_int_t       *fshi;

436     fsh_prepare_fsrec(vfsp);

438     if ((handle = id_alloc(fsh_idspace)) == -1)
439         return (-1);

441     fshi = kmem_alloc(sizeof (*fshi), KM_SLEEP);
442     mutex_init(&fshi->fshi_lock, NULL, MUTEX_DRIVER, NULL);
443     (void) memcpy(&fshi->fshi_hooks, hooks, sizeof (fshi->fshi_hooks));
444     fshi->fshi_handle = handle;
445     fshi->fshi_doomed = 0;
446     fshi->fshi_ref = 1;
447     fshi->fshi_vfsp = vfs;

449     mutex_enter(&fsh_lock);
450     rw_enter(&vfsp->vfs_fshrecord->fshfsr_lock, RW_WRITER);
451     list_insert_head(&vfsp->vfs_fshrecord->fshfsr_list, fshi);
452     rw_exit(&vfsp->vfs_fshrecord->fshfsr_lock);

454     list_insert_head(&fsh_map, fshi);
455     mutex_exit(&fsh_lock);

457     return (handle);

```

```

458 }

460 static int
461 fshi_hold(fsh_int_t *fshi)
462 {
463     int can_hold;

465     mutex_enter(&fshi->fshi_lock);
466     if (fshi->fshi_doomed == 1) {
467         can_hold = 0;
468     } else {
469         fshi->fshi_ref++;
470         can_hold = 1;
471     }
472     mutex_exit(&fshi->fshi_lock);

474     return (can_hold);
475 }

477 /*
478  * This function must not be called while fshfsr_lock is held. Doing so could
479  * cause a deadlock.
480  */
481 static void
482 fshi_rele(fsh_int_t *fshi)
483 {
484     int destroy;

486     mutex_enter(&fshi->fshi_lock);
487     ASSERT(fshi->fshi_ref > 0);
488     fshi->fshi_ref--;
489     if (fshi->fshi_ref == 0) {
490         ASSERT(fshi->fshi_doomed == 1);
491         destroy = 1;
492     } else {
493         destroy = 0;
494     }
495     mutex_exit(&fshi->fshi_lock);

497     if (destroy) {
498         /*
499          * At this point, we are sure that fsh_hook_remove() has been
500          * called, that's why we don't remove the fshi from fsh_map.
501          * fsh_hook_remove() did that already.
502          */
503         fsh_fsrecord_t *fsrecp;

505         if (fshi->fshi_hooks.remove_cb != NULL)
506             (*fshi->fshi_hooks.remove_cb)(
507                 fshi->fshi_hooks.arg, fshi->fshi_handle);
508         /*
509          * We don't have to call fsh_prepare_fsrec() here.
510          * fsh_fsrecord_t is already initialised, because we've found a
511          * mapping for the given handle.
512          */
513         fsrecp = fshi->fshi_vfsp->vfs_fshrecord;
514         ASSERT(fsrecp != NULL);
515         ASSERT(fsrecp != fsh_res_ptr);

517         rw_enter(&fsrecp->fshfsr_lock, RW_WRITER);
518         list_remove(&fsrecp->fshfsr_list, fshi);
519         rw_exit(&fsrecp->fshfsr_lock);

521         id_free(fsh_idspace, fshi->fshi_handle);
522         mutex_destroy(&fshi->fshi_lock);
523         kmem_free(fshi, sizeof (*fshi));

```

```

524     }
525 }

527 /*
528 * Used for removing a hook set.
529 *
530 * fsh_hook_remove() invalidates the given handle.
531 *
532 * It is guaranteed, that after successful return from fsh_hook_remove(),
533 * calls to vnodeops/vfsops, on the vfs_t on which the hook is installed, won't
534 * go through this hook.
535 *
536 * There is no guarantee that after fsh_hook_remove() returns, the hook
537 * associated with the handle won't be executing. Instead, it is guaranteed that
538 * when remove_cb() is called, the hook finished it's execution in all threads.
539 * It is safe to destroy all internal data associated with this hook inside
540 * remove_cb().
541 *
542 * It is possible that remove_cb() would be called before fsh_hook_remove()
543 * returns.
544 *
545 * Returns (-1) if hook wasn't found, 0 otherwise.
546 */
547 int
548 fsh_hook_remove(fsh_handle_t handle)
549 {
550     fsh_int_t     *fshi;

552     mutex_enter(&fsh_lock);
553     for (fshi = list_head(&fsh_map); fshi != NULL;
554          fshi = list_next(&fsh_map, fshi)) {
555         if (fshi->fshi_handle == handle) {
556             list_remove(&fsh_map, fshi);
557             break;
558         }
559     }

561     if (fshi == NULL)
562         return (-1);

564     mutex_enter(&fshi->fshi_lock);
565     ASSERT(fshi->fshi_doomed == 0);
566     fshi->fshi_doomed = 1;
567     mutex_exit(&fshi->fshi_lock);
568     mutex_exit(&fsh_lock);

570     fshi_rele(fshi);

572     return (0);
573 }

575 /*
576 * API for installing global mount/free callbacks.
577 *
578 * fsh_callback_t fields:
579 * fshc_arg - argument passed to the callbacks
580 * fshc_free - callback fired before VFS_FREEVFS() is called, after vfs_count
581 *           drops to 0
582 * fshc_mount - callback fired right before returning from domount()
583 * The first argument of these callbacks is the vfs_t that is mounted/freed.
584 * The second one is the fshc_arg.
585 *
586 * fsh_callback_handle_t is filled out by this function.
587 *
588 * This function must NOT be called in a callback, because it will cause
589 * a deadlock.

```

```

590 *
591 * Returns (-1) if hook/callback limit exceeded.
592 */
593 fsh_callback_handle_t
594 fsh_callback_install(fsh_callback_t *callback)
595 {
596     fsh_callback_int_t *fshci;
597     fsh_callback_handle_t handle;

599     if ((handle = id_alloc(fsh_idspace)) == -1)
600         return (-1);

602     fshci = (fsh_callback_int_t *)kmem_alloc(sizeof (*fshci), KM_SLEEP);
603     (void) memcpy(&fshci->fshci_cb, callback, sizeof (fshci->fshci_cb));
604     fshci->fshci_handle = handle;

606     /* If it is called in a {mount,free} callback, causes deadlock. */
607     rw_enter(&fsh_cblst_lock, RW_WRITER);
608     list_insert_head(&fsh_cblst, fshci);
609     rw_exit(&fsh_cblst_lock);

611     return (handle);
612 }

614 /*
615 * API for removing global mount/free callbacks.
616 *
617 * This function must NOT be called in a callback, because it will cause
618 * a deadlock.
619 *
620 * Returns (-1) if callback wasn't found, 0 otherwise.
621 */
622 int
623 fsh_callback_remove(fsh_callback_handle_t handle)
624 {
625     fsh_callback_int_t *fshci;

627     /* If it is called in a {mount,free} callback, causes deadlock. */
628     rw_enter(&fsh_cblst_lock, RW_WRITER);
629     for (fshci = list_head(&fsh_cblst); fshci != NULL;
630          fshci = list_next(&fsh_cblst, fshci)) {
631         if (fshci->fshci_handle == handle) {
632             list_remove(&fsh_cblst, fshci);
633             break;
634         }
635     }
636     rw_exit(&fsh_cblst_lock);

638     if (fshci == NULL)
639         return (-1);

641     kmem_free(fshci, sizeof (*fshci));
642     id_free(fsh_idspace, handle);

644     return (0);
645 }

647 /*
648 * This function is executed right before returning from domount()@vfs.c.
649 * We are sure that it's called only after fsh_init().
650 * It executes all the mount callbacks installed in the fsh.
651 *
652 * Since fsh_exec_mount_callbacks() is called only inside domount(), it is legal
653 * to call fsh_hook_{install,remove}() inside a mount callback WITHOUT holding
654 * this vfs_t. This guarantee should be preserved, because it's in the "Usage"
655 * section in the big theory statement at the top of this file.

```

```

656 */
657 void
658 fsh_exec_mount_callbacks(vfs_t *vfsp)
659 {
660     fsh_callback_int_t *fshci;
661     fsh_callback_t *cb;

663     rw_enter(&fsh_cblst_lock, RW_READER);
664     for (fshci = list_head(&fsh_cblst); fshci != NULL;
665          fshci = list_next(&fsh_cblst, fshci)) {
666         cb = &fshci->fshci_cb;
667         if (cb->fshc_mount != NULL)
668             (*(cb->fshc_mount))(vfsp, cb->fshc_arg);
669     }
670     rw_exit(&fsh_cblst_lock);
671 }

673 /*
674  * This function is executed right before VFS_FREEVFS() is called in
675  * vfs_rele()@vfs.c. We are sure that it's called only after fsh_init().
676  * It executes all the free callbacks installed in the fsh.
677  *
678  * free() callback is the point after the handles associated with the hooks
679  * installed on this vfs_t become invalid
680  */
681 void
682 fsh_exec_free_callbacks(vfs_t *vfsp)
683 {
684     fsh_callback_int_t *fshci;
685     fsh_callback_t *cb;

687     rw_enter(&fsh_cblst_lock, RW_READER);
688     for (fshci = list_head(&fsh_cblst); fshci != NULL;
689          fshci = list_next(&fsh_cblst, fshci)) {
690         cb = &fshci->fshci_cb;
691         if (cb->fshc_free != NULL)
692             (*(cb->fshc_free))(vfsp, cb->fshc_arg);
693     }
694     rw_exit(&fsh_cblst_lock);
695 }

697 /*
698  * API for vnode.c/vfs.c to start executing the fsh for a given operation.
699  *
700  * fsh_xxx() tries to find the first non-NULL xxx hook on the fshfsr_list. If it
701  * does, it executes it. If not, underlying vnodeop/vfsop is called.
702  *
703  * These interfaces are using fsh_res_ptr (in fsh_prepare_fsrec()), so it's
704  * absolutely necessary to call fsh_init() before using them. That's done in
705  * vfsinit().
706  *
707  * While these functions are executing, it's expected that necessary vfs_t's
708  * are held so that vfs_free() isn't called. vfs_free() expects that noone
709  * accesses vfs_fshrecord of a given vfs_t.
710  * It's also the caller's responsibility to keep vnode_t passed to fsh_foo()
711  * alive and valid.
712  * All these expectations are met because these functions are used only in
713  * corresponding {fop,fsop}_foo() functions.
714  */
715 int
716 fsh_read(vnode_t *vp, uio_t *uiop, int ioflag, cred_t *cr,
717          caller_context_t *ct)
718 {
719     int ret;
720     fsh_fsrecord_t *fsrecp;
721     fsh_int_t *fshi;

```

```

723     fsh_prepare_fsrec(vp->v_vfsp);
724     fsrecp = vp->v_vfsp->vfs_fshrecord;

726     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
727     if (!(fsrecp->fshfsr_enabled)) {
728         rw_exit(&fsrecp->fshfsr_lock);
729         return ((*vp->v_op->vop_read))(vp, uiop, ioflag, cr, ct);
730     }

732     for (fshi = list_head(&fsrecp->fshfsr_list); fshi != NULL;
733          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
734         if (fshi->fshi_hooks.read != NULL)
735             if (fshi_hold(fshi))
736                 break;
737     }
738     rw_exit(&fsrecp->fshfsr_lock);

740     if (fshi == NULL)
741         return ((*vp->v_op->vop_read))(vp, uiop, ioflag, cr, ct);

743     ret = (*fshi->fshi_hooks.read)(fshi, fshi->fshi_hooks.arg,
744                                   vp, uiop, ioflag, cr, ct);
745     fshi_rele(fshi);
746     return (ret);
747 }

749 int
750 fsh_write(vnode_t *vp, uio_t *uiop, int ioflag, cred_t *cr,
751           caller_context_t *ct)
752 {
753     fsh_int_t *fshi;
754     int ret;
755     fsh_fsrecord_t *fsrecp;

757     fsh_prepare_fsrec(vp->v_vfsp);
758     fsrecp = vp->v_vfsp->vfs_fshrecord;

760     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
761     if (!(vp->v_vfsp->vfs_fshrecord->fshfsr_enabled)) {
762         rw_exit(&fsrecp->fshfsr_lock);
763         return ((*vp->v_op->vop_write))(vp, uiop, ioflag, cr, ct);
764     }

766     for (fshi = list_head(&fsrecp->fshfsr_list); fshi != NULL;
767          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
768         if (fshi->fshi_hooks.write != NULL)
769             if (fshi_hold(fshi))
770                 break;
771     }
772     rw_exit(&fsrecp->fshfsr_lock);

774     if (fshi == NULL)
775         return ((*vp->v_op->vop_write))(vp, uiop, ioflag, cr, ct);

777     ret = (*fshi->fshi_hooks.write)(fshi, fshi->fshi_hooks.arg,
778                                   vp, uiop, ioflag, cr, ct);
779     fshi_rele(fshi);
780     return (ret);
781 }

783 int
784 fsh_mount(vfs_t *vfsp, vnode_t *mvp, struct mounta *uap, cred_t *cr)
785 {
786     fsh_fsrecord_t *fsrecp;
787     fsh_int_t *fshi;

```

```

788     int ret;
790     fsh_prepare_fsrec(vfsp);
791     fsrecp = vfs->vfs_fshrecord;

793     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
794     if (!(fsrecp->fshfsr_enabled)) {
795         rw_exit(&fsrecp->fshfsr_lock);
796         return ((*vfs->vfs_op->vfs_mount)(vfs, mvp, uap, cr));
797     }

799     for (fshi = list_head(&fsrecp->fshfsr_list); fshi != NULL;
800          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
801         if (fshi->fshi_hooks.mount != NULL)
802             if (fshi_hold(fshi))
803                 break;
804     }
805     rw_exit(&fsrecp->fshfsr_lock);

807     if (fshi == NULL)
808         return ((*vfs->vfs_op->vfs_mount)(vfs, mvp, uap, cr));

810     ret = (*fshi->fshi_hooks.mount)(fshi, fshi->fshi_hooks.arg,
811                                   vfs, mvp, uap, cr);
812     fshi_rele(fshi);
813     return (ret);
814 }

816 int
817 fsh_unmount(vfs_t *vfs, int flag, cred_t *cr)
818 {
819     fsh_fsrecord_t *fsrecp;
820     fsh_int_t *fshi;
821     int ret;

823     fsh_prepare_fsrec(vfsp);
824     fsrecp = vfs->vfs_fshrecord;

826     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
827     if (!(fsrecp->fshfsr_enabled)) {
828         rw_exit(&fsrecp->fshfsr_lock);
829         return ((*vfs->vfs_op->vfs_unmount)(vfs, flag, cr));
830     }

832     for (fshi = list_head(&fsrecp->fshfsr_list); fshi != NULL;
833          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
834         if (fshi->fshi_hooks.unmount != NULL)
835             if (fshi_hold(fshi))
836                 break;
837     }
838     rw_exit(&fsrecp->fshfsr_lock);

840     if (fshi == NULL)
841         return ((*vfs->vfs_op->vfs_unmount)(vfs, flag, cr));

843     ret = (*fshi->fshi_hooks.unmount)(fshi, fshi->fshi_hooks.arg,
844                                       vfs, flag, cr);
845     fshi_rele(fshi);
846     return (ret);
847 }

849 /*
850  * This is the funtion used by fsh_prepare_fsrec() to allocate a new
851  * fsh_fsrecord. This function is called by the first function which
852  * access the vfs_fshrecord and finds out it's NULL.
853  */

```

```

854 static fsh_fsrecord_t *
855 fsh_fsrec_create()
856 {
857     fsh_fsrecord_t *fsrecp;

859     fsrecp = (fsh_fsrecord_t *)kmem_zalloc(sizeof (*fsrecp), KM_SLEEP);
860     list_create(&fsrecp->fshfsr_list, sizeof (fsh_int_t),
861               offsetof(fsh_int_t, fshi_next));
862     rw_init(&fsrecp->fshfsr_lock, NULL, RW_DRIVER, NULL);
863     fsrecp->fshfsr_enabled = 1;
864     return (fsrecp);
865 }

868 /*
869  * This call can be used ONLY in vfs_free(). It's assumed that no other
870  * fsh calls using the vfs_t that owns the fsh_fsrecord to be destroyed
871  * are executing while a call to fsh_fsrec_destroy() is made. With this
872  * assumptions, no concurrency issues occur.
873  *
874  * Before calling this function outside the fsh, it's sufficient and
875  * required to check if the passed fsh_fsrecord * is not NULL. We don't
876  * have to check if it is not equal to fsh_res_ptr, because all the fsh API
877  * calls involving this vfs_t should end before vfs_free() is called
878  * (outside the fsh, fsh_fsrecord is never equal to fsh_res_ptr). That is
879  * guaranteed by the explicit requirement that the caller of fsh API holds
880  * the vfs_t when needed.
881  *
882  * All the remaining hooks are being removed.
883  */
884 void
885 fsh_fsrec_destroy(struct fsh_fsrecord *volatile fsrecp)
886 {
887     fsh_int_t *fshi;

889     VERIFY(fsrecp != NULL);

891     _NOTE(CONSTCOND)
892     while (1) {
893         mutex_enter(&fsh_lock);
894         /* No need here to hold fshfsr_lock */
895         fshi = list_remove_head(&fsrecp->fshfsr_list);
896         if (fshi == NULL) {
897             mutex_exit(&fsh_lock);
898             break;
899         }
900         ASSERT(fshi->fshi_doomed == 0);
901         list_remove(&fsh_map, fshi);
902         mutex_exit(&fsh_lock);

904         if (fshi->fshi_hooks.remove_cb != NULL)
905             (*fshi->fshi_hooks.remove_cb)(fshi->fshi_hooks.arg,
906                                         fshi->fshi_handle);
907         id_free(fsh_idspace, fshi->fshi_handle);
908         mutex_destroy(&fshi->fshi_lock);
909         kmem_free(fshi, sizeof (*fshi));

911     }

913     list_destroy(&fsrecp->fshfsr_list);
914     rw_destroy(&fsrecp->fshfsr_lock);
915     kmem_free(fsrecp, sizeof (*fsrecp));
916 }

918 /*
919  * fsh_init() is called in vfsinit()@vfs.c. This function MUST be called

```

```

920 * before every other fsh call.
921 */
922 void
923 fsh_init(void)
924 {
925     rw_init(&fsh_cblst_lock, NULL, RW_DRIVER, NULL);
926     list_create(&fsh_cblst, sizeof (fsh_callback_int_t),
927               offsetof(fsh_callback_int_t, fshci_next));
928
929     mutex_init(&fsh_lock, NULL, MUTEX_DRIVER, NULL);
930
931     list_create(&fsh_map, sizeof (fsh_int_t), offsetof(fsh_int_t,
932               fshi_global));
933
934     /* See comment above fsh_prepare_fsrec() */
935     fsh_res_ptr = (void *)-1;
936
937     fsh_idspace = id_space_create("fsh", 0, fsh_limit);
938 }
939
940 /*
941 * These functions are used to pass control to the next hook or underlying
942 * vop or vfsop. It's client doesn't have to worry about any locking.
943 */
944 int
945 fsh_next_read(fsh_int_t *fshi, vnode_t *vp, uio_t *uiop, int ioflag,
946              cred_t *cr, caller_context_t *ct)
947 {
948     int ret;
949     fsh_fsrecord_t *fsrecp = vp->v_vfsp->vfs_fshrecord;
950
951     /*
952     * The passed fshi is the previous hook (the one from which we've been
953     * called). We need to find the next one.
954     */
955     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
956     for (fshi = list_next(&fsrecp->fshfsr_list, fshi); fshi != NULL;
957          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
958         if (fshi->fshi_hooks.read != NULL)
959             if (fshi_hold(fshi))
960                 break;
961     }
962     rw_exit(&fsrecp->fshfsr_lock);
963
964     if (fshi == NULL)
965         return ((*vp->v_op->vop_read)(vp, uiop, ioflag, cr, ct));
966
967     ret = (*fshi->fshi_hooks.read)(fshi, fshi->fshi_hooks.arg,
968                                  vp, uiop, ioflag, cr, ct);
969     fshi_rele(fshi);
970     return (ret);
971 }
972
973 int
974 fsh_next_write(fsh_int_t *fshi, vnode_t *vp, uio_t *uiop, int ioflag,
975               cred_t *cr, caller_context_t *ct)
976 {
977     fsh_fsrecord_t *fsrecp = vp->v_vfsp->vfs_fshrecord;
978     int ret;
979
980     /*
981     * The passed fshi is the previous hook (the one from which we've been
982     * called). We need to find the next one.
983     */
984     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
985     for (fshi = list_next(&fsrecp->fshfsr_list, fshi); fshi != NULL;

```

```

986         fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
987             if (fshi->fshi_hooks.write != NULL)
988                 if (fshi_hold(fshi))
989                     break;
990         }
991     rw_exit(&fsrecp->fshfsr_lock);
992
993     if (fshi == NULL)
994         return ((*vp->v_op->vop_write)(vp, uiop, ioflag, cr, ct));
995
996     ret = (*fshi->fshi_hooks.write)(fshi, fshi->fshi_hooks.arg,
997                                   vp, uiop, ioflag, cr, ct);
998     fshi_rele(fshi);
999     return (ret);
1000 }
1001
1002 int
1003 fsh_next_mount(fsh_int_t *fshi, vfs_t *vfsp, vnode_t *mvp, struct mounta *uap,
1004               cred_t *cr)
1005 {
1006     fsh_fsrecord_t *fsrecp = vfsp->vfs_fshrecord;
1007     int ret;
1008
1009     /*
1010     * The passed fshi is the previous hook (the one from which we've been
1011     * called). We need to find the next one.
1012     */
1013     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
1014     for (fshi = list_next(&fsrecp->fshfsr_list, fshi); fshi != NULL;
1015          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
1016         if (fshi->fshi_hooks.mount != NULL)
1017             if (fshi_hold(fshi))
1018                 break;
1019     }
1020     rw_exit(&fsrecp->fshfsr_lock);
1021
1022     if (fshi == NULL)
1023         return ((*vfsp->vfs_op->vfs_mount)(vfsp, mvp, uap, cr));
1024
1025     ret = (*fshi->fshi_hooks.mount)(fshi, fshi->fshi_hooks.arg,
1026                                   vfsp, mvp, uap, cr);
1027     fshi_rele(fshi);
1028     return (ret);
1029 }
1030
1031 int
1032 fsh_next_unmount(fsh_int_t *fshi, vfs_t *vfsp, int flag, cred_t *cr)
1033 {
1034     fsh_fsrecord_t *fsrecp = vfsp->vfs_fshrecord;
1035     int ret;
1036
1037     /*
1038     * The passed fshi is the previous hook (the one from which we've been
1039     * called). We need to find the next one.
1040     */
1041     rw_enter(&fsrecp->fshfsr_lock, RW_READER);
1042     for (fshi = list_next(&fsrecp->fshfsr_list, fshi); fshi != NULL;
1043          fshi = list_next(&fsrecp->fshfsr_list, fshi)) {
1044         if (fshi->fshi_hooks.unmount != NULL)
1045             if (fshi_hold(fshi))
1046                 break;
1047     }
1048     rw_exit(&fsrecp->fshfsr_lock);
1049
1050     if (fshi == NULL)
1051         return ((*vfsp->vfs_op->vfs_unmount)(vfsp, flag, cr));

```



```
1053     ret = (*fshi->fshi_hooks.unmount)(fshi, fshi->fshi_hooks.arg,  
1054         vfsp, flag, cr);  
1055     fshi_rele(fshi);  
1056     return (ret);  
1057 }
```

new/usr/src/uts/common/fs/vfs.c

1

```
*****
118568 Tue Sep 3 18:09:41 2013
new/usr/src/uts/common/fs/vfs.c
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22 * Copyright (c) 1988, 2010, Oracle and/or its affiliates. All rights reserved.
23 */

25 /*      Copyright (c) 1983, 1984, 1985, 1986, 1987, 1988, 1989 AT&T      */
26 /*      All Rights Reserved      */

28 /*
29 * University Copyright- Copyright (c) 1982, 1986, 1988
30 * The Regents of the University of California
31 * All Rights Reserved
32 *
33 * University Acknowledgment- Portions of this document are derived from
34 * software developed by the University of California, Berkeley, and its
35 * contributors.
36 */

38 #include <sys/types.h>
39 #include <sys/t_lock.h>
40 #include <sys/param.h>
41 #include <sys/errno.h>
42 #include <sys/user.h>
43 #include <sys/fstyp.h>
44 #include <sys/kmem.h>
45 #include <sys/system.h>
46 #include <sys/proc.h>
47 #include <sys/mount.h>
48 #include <sys/vfs.h>
49 #include <sys/vfs_opreg.h>
50 #include <sys/fem.h>
51 #include <sys/mntent.h>
52 #include <sys/stat.h>
53 #include <sys/statvfs.h>
54 #include <sys/statfs.h>
55 #include <sys/cred.h>
56 #include <sys/vnode.h>
57 #include <sys/rwstlock.h>
58 #include <sys/dnld.h>
59 #include <sys/file.h>
60 #include <sys/time.h>
61 #include <sys/atomic.h>
```

new/usr/src/uts/common/fs/vfs.c

2

```
62 #include <sys/cmn_err.h>
63 #include <sys/buf.h>
64 #include <sys/swap.h>
65 #include <sys/debug.h>
66 #include <sys/vnode.h>
67 #include <sys/modctl.h>
68 #include <sys/ddi.h>
69 #include <sys/pathname.h>
70 #include <sys/bootconf.h>
71 #include <sys/dumphdr.h>
72 #include <sys/dc_ki.h>
73 #include <sys/poll.h>
74 #include <sys/sunddi.h>
75 #include <sys/sysmacros.h>
76 #include <sys/zone.h>
77 #include <sys/policy.h>
78 #include <sys/ctfs.h>
79 #include <sys/objfs.h>
80 #include <sys/console.h>
81 #include <sys/reboot.h>
82 #include <sys/attr.h>
83 #include <sys/zio.h>
84 #include <sys/spa.h>
85 #include <sys/lofi.h>
86 #include <sys/bootprops.h>
87 #include <sys/fsh.h>
88 #include <sys/fsh_impl.h>

90 #include <vm/page.h>

92 #include <fs/fs_subr.h>
93 /* Private interfaces to create vopstats-related data structures */
94 extern void initialize_vopstats(vopstats_t *);
95 extern vopstats_t *get_fstype_vopstats(struct vfs *, struct vfssw *);
96 extern vsk_anchor_t *get_vskstat_anchor(struct vfs *);

98 static void vfs_clearmntopt_nolock(mntopts_t *, const char *, int);
99 static void vfs_setmntopt_nolock(mntopts_t *, const char *,
100 const char *, int, int);
101 static int vfs_optionisset_nolock(const mntopts_t *, const char *, char **);
102 static void vfs_freemnttab(struct vfs *);
103 static void vfs_freeopt(mntopt_t *);
104 static void vfs_swapopttbl_nolock(mntopts_t *, mntopts_t *);
105 static void vfs_swapopttbl(mntopts_t *, mntopts_t *);
106 static void vfs_copyopttbl_extend(const mntopts_t *, mntopts_t *, int);
107 static void vfs_createopttbl_extend(mntopts_t *, const char *,
108 const mntopts_t *);
109 static char **vfs_copycancelopt_extend(char **const, int);
110 static void vfs_freecancelopt(char **);
111 static void getrootfs(char **, char **);
112 static int getmacpath(dev_info_t *, void *);
113 static void vfs_mnttabvop_setup(void);

115 struct ipmnt {
116     struct ipmnt *mip_next;
117     dev_t mip_dev;
118     struct vfs *mip_vfsp;
119 };

----- unchanged portion omitted -----

214 /*
215  * File system operation dispatch functions.
216  */

218 int
219 fsop_mount(vfs_t *vfsp, vnode_t *mvp, struct mounta *uap, cred_t *cr)
```

```

220 {
221     return (fsh_mount(vfsp, mvp, uap, cr));
219     return (*(vfs_op->vfs_mount)(vfs, mvp, uap, cr);
222 }

224 int
225 fsop_unmount(vfs_t *vfs, int flag, cred_t *cr)
226 {
227     return (fsh_unmount(vfsp, flag, cr));
225     return (*(vfs_op->vfs_unmount)(vfs, flag, cr);
228 }

    unchanged_portion_omitted

1085 /*
1086 * Common mount code. Called from the system call entry point, from autofs,
1087 * nfsv4 trigger mounts, and from pxfs.
1088 *
1089 * Takes the effective file system type, mount arguments, the mount point
1090 * vnode, flags specifying whether the mount is a remount and whether it
1091 * should be entered into the vfs list, and credentials. Fills in its vfsp
1092 * parameter with the mounted file system instance's vfs.
1093 *
1094 * Note that the effective file system type is specified as a string. It may
1095 * be null, in which case it's determined from the mount arguments, and may
1096 * differ from the type specified in the mount arguments; this is a hook to
1097 * allow interposition when instantiating file system instances.
1098 *
1099 * The caller is responsible for releasing its own hold on the mount point
1100 * vp (this routine does its own hold when necessary).
1101 * Also note that for remounts, the mount point vp should be the vnode for
1102 * the root of the file system rather than the vnode that the file system
1103 * is mounted on top of.
1104 */
1105 int
1106 domount(char *fsname, struct mounta *uap, vnode_t *vp, struct cred *credp,
1107         struct vfs **vfsp)
1108 {
1109     struct vfsw      *vsw;
1110     vfsops_t        *vfops;
1111     struct vfs      *vfsp;
1112     struct vnode    *bvp;
1113     dev_t           bdev = 0;
1114     mnt_mntopts_t  mntopts;
1115     int             error = 0;
1116     int             copyout_error = 0;
1117     int             ovflags;
1118     char            *opts = uap->optptr;
1119     char            *inargs = opts;
1120     int             optlen = uap->optlen;
1121     int             remount;
1122     int             rdonly;
1123     int             nbmand = 0;
1124     int             delmip = 0;
1125     int             addmip = 0;
1126     int             splice = ((uap->flags & MS_NOSPLICE) == 0);
1127     int             fromspace = (uap->flags & MS_SYSSPACE) ?
1128         UIO_SYSSPACE : UIO_USERSPACE;
1129     char            *resource = NULL, *mountpt = NULL;
1130     refstr_t        *oldresource, *oldmntpt;
1131     struct pathname pn, rpn;
1132     vsk_anchor_t    *vskap;
1133     char fstname[FSTYPSZ];

1135 /*
1136 * The v_flag value for the mount point vp is permanently set
1137 * to VVFSLOCK so that no one bypasses the vn_vfs*locks routine

```

```

1138     * for mount point locking.
1139     */
1140     mutex_enter(&vp->v_lock);
1141     vp->v_flag |= VVFSLOCK;
1142     mutex_exit(&vp->v_lock);

1144     mnt_mntopts.mo_count = 0;
1145     /*
1146     * Find the ops vector to use to invoke the file system-specific mount
1147     * method. If the fsname argument is non-NULL, use it directly.
1148     * Otherwise, dig the file system type information out of the mount
1149     * arguments.
1150     *
1151     * A side effect is to hold the vfsw entry.
1152     *
1153     * Mount arguments can be specified in several ways, which are
1154     * distinguished by flag bit settings. The preferred way is to set
1155     * MS_OPTIONSTR, indicating an 8 argument mount with the file system
1156     * type supplied as a character string and the last two arguments
1157     * being a pointer to a character buffer and the size of the buffer.
1158     * On entry, the buffer holds a null terminated list of options; on
1159     * return, the string is the list of options the file system
1160     * recognized. If MS_DATA is set arguments five and six point to a
1161     * block of binary data which the file system interprets.
1162     * A further wrinkle is that some callers don't set MS_FSS and MS_DATA
1163     * consistently with these conventions. To handle them, we check to
1164     * see whether the pointer to the file system name has a numeric value
1165     * less than 256. If so, we treat it as an index.
1166     */
1167     if (fsname != NULL) {
1168         if ((vsw = vfs_getvfsw(fsname)) == NULL) {
1169             return (EINVAL);
1170         }
1171     } else if (uap->flags & (MS_OPTIONSTR | MS_DATA | MS_FSS)) {
1172         size_t n;
1173         uint_t fstype;

1175         fsname = fstname;

1177         if ((fstype = (uintptr_t)uap->fstype) < 256) {
1178             RLOCK_VFSSW();
1179             if (fstype == 0 || fstype >= nfstype ||
1180                 !ALLOCATED_VFSSW(&vfsw[fstype])) {
1181                 RUNLOCK_VFSSW();
1182                 return (EINVAL);
1183             }
1184             (void) strcpy(fsname, vfsw[fstype].vsw_name);
1185             RUNLOCK_VFSSW();
1186             if ((vsw = vfs_getvfsw(fsname)) == NULL)
1187                 return (EINVAL);
1188         } else {
1189             /*
1190             * Handle either kernel or user address space.
1191             */
1192             if (uap->flags & MS_SYSSPACE) {
1193                 error = copystr(uap->fstype, fsname,
1194                               FSTYPSZ, &n);
1195             } else {
1196                 error = copyinstr(uap->fstype, fsname,
1197                                 FSTYPSZ, &n);
1198             }
1199             if (error) {
1200                 if (error == ENAMETOOLONG)
1201                     return (EINVAL);
1202                 return (error);
1203             }

```

```

1204         if ((vswp = vfs_getvfsw(fsname)) == NULL)
1205             return (EINVAL);
1206     }
1207 } else {
1208     if ((vswp = vfs_getvfswbyvfswops(vfs_getops(rootvfs))) == NULL)
1209         return (EINVAL);
1210     fsname = vswp->vsw_name;
1211 }
1212 if (!VFS_INSTALLED(vswp))
1213     return (EINVAL);
1214
1215 if ((error = secpolicy_fs_allowed_mount(fsname)) != 0) {
1216     vfs_unrefvfsw(vswp);
1217     return (error);
1218 }
1219
1220 vfsops = &vswp->vsw_vfsops;
1221
1222 vfs_copyopttbl(&vswp->vsw_optproto, &mnt_mntopts);
1223 /*
1224  * Fetch mount options and parse them for generic vfs options
1225  */
1226 if (uap->flags & MS_OPTIONSTR) {
1227     /*
1228      * Limit the buffer size
1229      */
1230     if (optlen < 0 || optlen > MAX_MNTOPT_STR) {
1231         error = EINVAL;
1232         goto errout;
1233     }
1234     if ((uap->flags & MS_SYSSPACE) == 0) {
1235         inargs = kmem_alloc(MAX_MNTOPT_STR, KM_SLEEP);
1236         inargs[0] = '\0';
1237         if (optlen) {
1238             error = copyinstr(opts, inargs, (size_t)optlen,
1239                 NULL);
1240             if (error) {
1241                 goto errout;
1242             }
1243         }
1244     }
1245     vfs_parsemntopts(&mnt_mntopts, inargs, 0);
1246 }
1247 /*
1248  * Flag bits override the options string.
1249  */
1250 if (uap->flags & MS_REMOUNT)
1251     vfs_setmntopt_nolock(&mnt_mntopts, MNTOPT_REMOUNT, NULL, 0, 0);
1252 if (uap->flags & MS_RDONLY)
1253     vfs_setmntopt_nolock(&mnt_mntopts, MNTOPT_RO, NULL, 0, 0);
1254 if (uap->flags & MS_NOSUID)
1255     vfs_setmntopt_nolock(&mnt_mntopts, MNTOPT_NOSUID, NULL, 0, 0);
1256
1257 /*
1258  * Check if this is a remount; must be set in the option string and
1259  * the file system must support a remount option.
1260  */
1261 if (remount = vfs_optionisset_nolock(&mnt_mntopts,
1262     MNTOPT_REMOUNT, NULL)) {
1263     if (!(vswp->vsw_flag & VSW_CANREMOUNT)) {
1264         error = ENOTSUP;
1265         goto errout;
1266     }
1267     uap->flags |= MS_REMOUNT;
1268 }

```

```

1270     /*
1271      * uap->flags and vfs_optionisset() should agree.
1272      */
1273     if (rdonly = vfs_optionisset_nolock(&mnt_mntopts, MNTOPT_RO, NULL)) {
1274         uap->flags |= MS_RDONLY;
1275     }
1276     if (vfs_optionisset_nolock(&mnt_mntopts, MNTOPT_NOSUID, NULL)) {
1277         uap->flags |= MS_NOSUID;
1278     }
1279     nbmand = vfs_optionisset_nolock(&mnt_mntopts, MNTOPT_NBMAND, NULL);
1280     ASSERT(splice || !remount);
1281     /*
1282      * If we are splicing the fs into the namespace,
1283      * perform mount point checks.
1284      *
1285      * We want to resolve the path for the mount point to eliminate
1286      * '.' and '..' and symlinks in mount points; we can't do the
1287      * same for the resource string, since it would turn
1288      * "/dev/dsk/c0t0d0s0" into "/devices/pci@...". We need to do
1289      * this before grabbing vn_vfswlock(), because otherwise we
1290      * would deadlock with lookuppn().
1291      */
1292     if (splice) {
1293         ASSERT(vp->v_count > 0);
1294
1295         /*
1296          * Pick up mount point and device from appropriate space.
1297          */
1298         if (pn_get(uap->spec, fromspace, &pn) == 0) {
1299             resource = kmem_alloc(pn.pn_pathlen + 1,
1300                 KM_SLEEP);
1301             (void) strcpy(resource, pn.pn_path);
1302             pn_free(&pn);
1303         }
1304         /*
1305          * Do a lookupname prior to taking the
1306          * writelock. Mark this as completed if
1307          * successful for later cleanup and addition to
1308          * the mount in progress table.
1309          */
1310         if ((uap->flags & MS_GLOBAL) == 0 &&
1311             lookupname(uap->spec, fromspace,
1312                 FOLLOW, NULL, &bpv) == 0) {
1313             admip = 1;
1314         }
1315
1316         if ((error = pn_get(uap->dir, fromspace, &pn)) == 0) {
1317             pathname_t *pnp;
1318
1319             if (*pn.pn_path != '/') {
1320                 error = EINVAL;
1321                 pn_free(&pn);
1322                 goto errout;
1323             }
1324             pn_alloc(&rpnp);
1325             /*
1326              * Kludge to prevent autofs from deadlocking with
1327              * itself when it calls domount().
1328              *
1329              * If autofs is calling, it is because it is doing
1330              * (autofs) mounts in the process of an NFS mount. A
1331              * lookuppn() here would cause us to block waiting for
1332              * said NFS mount to complete, which can't since this
1333              * is the thread that was supposed to do it.
1334              */
1335             if (fromspace == UIO_USERSPACE) {

```

```

1336         if ((error = lookuppn(&pn, &rp, FOLLOW, NULL,
1337             NULL)) == 0) {
1338             pnp = &rp;
1339         } else {
1340             /*
1341              * The file disappeared or otherwise
1342              * became inaccessible since we opened
1343              * it; might as well fail the mount
1344              * since the mount point is no longer
1345              * accessible.
1346              */
1347             pn_free(&rp);
1348             pn_free(&pn);
1349             goto errout;
1350         }
1351     } else {
1352         pnp = &pn;
1353     }
1354     mountpt = kmem_alloc(pnp->pn_pathlen + 1, KM_SLEEP);
1355     (void) strcpy(mountpt, pnp->pn_path);
1356
1357     /*
1358      * If the addition of the zone's rootpath
1359      * would push us over a total path length
1360      * of MAXPATHLEN, we fail the mount with
1361      * ENAMETOOLONG, which is what we would have
1362      * gotten if we were trying to perform the same
1363      * mount in the global zone.
1364      *
1365      * strlen() doesn't count the trailing
1366      * '\0', but zone_rootpathlen counts both a
1367      * trailing '/' and the terminating '\0'.
1368      */
1369     if (((curproc->p_zone->zone_rootpathlen - 1 +
1370         strlen(mountpt)) > MAXPATHLEN ||
1371         (resource != NULL &&
1372          (curproc->p_zone->zone_rootpathlen - 1 +
1373           strlen(resource)) > MAXPATHLEN)) {
1374         error = ENAMETOOLONG;
1375     }
1376
1377     pn_free(&rp);
1378     pn_free(&pn);
1379 }
1380
1381 if (error)
1382     goto errout;
1383
1384 /*
1385  * Prevent path name resolution from proceeding past
1386  * the mount point.
1387  */
1388 if (vn_vfswlock(vp) != 0) {
1389     error = EBUSY;
1390     goto errout;
1391 }
1392
1393 /*
1394  * Verify that it's legitimate to establish a mount on
1395  * the prospective mount point.
1396  */
1397 if (vn_mountedvfs(vp) != NULL) {
1398     /*
1399      * The mount point lock was obtained after some
1400      * other thread raced through and established a mount.
1401      */

```

```

1402         vn_vfsunlock(vp);
1403         error = EBUSY;
1404         goto errout;
1405     }
1406     if (vp->v_flag & VNOMOUNT) {
1407         vn_vfsunlock(vp);
1408         error = EINVAL;
1409         goto errout;
1410     }
1411 }
1412 if ((uap->flags & (MS_DATA | MS_OPTIONSTR)) == 0) {
1413     uap->dataptr = NULL;
1414     uap->datalen = 0;
1415 }
1416
1417 /*
1418  * If this is a remount, we don't want to create a new VFS.
1419  * Instead, we pass the existing one with a remount flag.
1420  */
1421 if (remount) {
1422     /*
1423      * Confirm that the mount point is the root vnode of the
1424      * file system that is being remounted.
1425      * This can happen if the user specifies a different
1426      * mount point directory pathname in the (re)mount command.
1427      *
1428      * Code below can only be reached if splice is true, so it's
1429      * safe to do vn_vfsunlock() here.
1430      */
1431     if ((vp->v_flag & VROOT) == 0) {
1432         vn_vfsunlock(vp);
1433         error = ENOENT;
1434         goto errout;
1435     }
1436     /*
1437      * Disallow making file systems read-only unless file system
1438      * explicitly allows it in its vfssw. Ignore other flags.
1439      */
1440     if (rdonly && vn_is_readonly(vp) == 0 &&
1441         (vswp->vsw_flag & VSW_CANRWRO) == 0) {
1442         vn_vfsunlock(vp);
1443         error = EINVAL;
1444         goto errout;
1445     }
1446     /*
1447      * Disallow changing the NBMAND disposition of the file
1448      * system on remounts.
1449      */
1450     if ((nbmand && ((vp->v_vfsp->vfs_flag & VFS_NBMAND) == 0)) ||
1451         (!nbmand && (vp->v_vfsp->vfs_flag & VFS_NBMAND))) {
1452         vn_vfsunlock(vp);
1453         error = EINVAL;
1454         goto errout;
1455     }
1456     vfsp = vp->v_vfsp;
1457     ovflags = vfsp->vfs_flag;
1458     vfsp->vfs_flag |= VFS_REMOUNT;
1459     vfsp->vfs_flag &= ~VFS_RDONLY;
1460 } else {
1461     vfsp = vfs_alloc(KM_SLEEP);
1462     VFS_INIT(vfsp, vfsops, NULL);
1463 }
1464
1465 VFS_HOLD(vfsp);
1466
1467 if ((error = lofi_add(fsname, vfsp, &mnt_mntopts, uap)) != 0) {

```

```

1468         if (!remount) {
1469             if (splice)
1470                 vn_vfsunlock(vp);
1471             vfs_free(vfsp);
1472         } else {
1473             vn_vfsunlock(vp);
1474             VFS_RELE(vfsp);
1475         }
1476         goto errout;
1477     }
1479     /*
1480     * PRIV_SYS_MOUNT doesn't mean you can become root.
1481     */
1482     if (vfsp->vfs_lofi_minor != 0) {
1483         uap->flags |= MS_NOSUID;
1484         vfs_setmntopt_nolock(&mnt_mntopts, MNTOPT_NOSUID, NULL, 0, 0);
1485     }
1487     /*
1488     * The vfs_reflock is not used anymore the code below explicitly
1489     * holds it preventing others accessing it directly.
1490     */
1491     if ((sema_tryv(&vfsp->vfs_reflock) == 0) &&
1492         !(vfsp->vfs_flag & VFS_REMOUNT))
1493         cmn_err(CE_WARN,
1494             "mount type %s couldn't get vfs_reflock", vswp->vsw_name);
1496     /*
1497     * Lock the vfs. If this is a remount we want to avoid spurious amount
1498     * failures that happen as a side-effect of fsflush() and other mount
1499     * and unmount operations that might be going on simultaneously and
1500     * may have locked the vfs currently. To not return EBUSY immediately
1501     * here we use vfs_lock_wait() instead vfs_lock() for the remount case.
1502     */
1503     if (!remount) {
1504         if (error = vfs_lock(vfsp)) {
1505             vfsp->vfs_flag = ovflags;
1507             lofi_remove(vfsp);
1509             if (splice)
1510                 vn_vfsunlock(vp);
1511             vfs_free(vfsp);
1512             goto errout;
1513         }
1514     } else {
1515         vfs_lock_wait(vfsp);
1516     }
1518     /*
1519     * Add device to mount in progress table, global mounts require special
1520     * handling. It is possible that we have already done the lookupname
1521     * on a spliced, non-global fs. If so, we don't want to do it again
1522     * since we cannot do a lookupname after taking the
1523     * wlock above. This case is for a non-spliced, non-global filesystem.
1524     */
1525     if (!addmip) {
1526         if ((uap->flags & MS_GLOBAL) == 0 &&
1527             lookupname(uap->spec, fromspace, FOLLOW, NULL, &bvp) == 0) {
1528             addmip = 1;
1529         }
1530     }
1532     if (addmip) {
1533         vnode_t *lvp = NULL;

```

```

1535         error = vfs_get_lofi(vfsp, &lvp);
1536         if (error > 0) {
1537             lofi_remove(vfsp);
1539             if (splice)
1540                 vn_vfsunlock(vp);
1541             vfs_unlock(vfsp);
1543             if (remount) {
1544                 VFS_RELE(vfsp);
1545             } else {
1546                 vfs_free(vfsp);
1547             }
1549             goto errout;
1550         } else if (error == -1) {
1551             bdev = bvp->v_rdev;
1552             VN_RELE(bvp);
1553         } else {
1554             bdev = lvp->v_rdev;
1555             VN_RELE(lvp);
1556             VN_RELE(bvp);
1557         }
1559         vfs_addmip(bdev, vfsp);
1560         addmip = 0;
1561         delmip = 1;
1562     }
1563     /*
1564     * Invalidate cached entry for the mount point.
1565     */
1566     if (splice)
1567         dnlc_purge_vp(vp);
1569     /*
1570     * If have an option string but the filesystem doesn't supply a
1571     * prototype options table, create a table with the global
1572     * options and sufficient room to accept all the options in the
1573     * string. Then parse the passed in option string
1574     * accepting all the options in the string. This gives us an
1575     * option table with all the proper cancel properties for the
1576     * global options.
1577     *
1578     * Filesystems that supply a prototype options table are handled
1579     * earlier in this function.
1580     */
1581     if (uap->flags & MS_OPTIONSTR) {
1582         if (!(vswp->vsw_flag & VSW_HASPROTO)) {
1583             mntopts_t tmp_mntopts;
1585             tmp_mntopts.mo_count = 0;
1586             vfs_createopttbl_extend(&tmp_mntopts, inargs,
1587                 &mnt_mntopts);
1588             vfs_parsemntopts(&tmp_mntopts, inargs, 1);
1589             vfs_swapopttbl_nolock(&mnt_mntopts, &tmp_mntopts);
1590             vfs_freeopttbl(&tmp_mntopts);
1591         }
1592     }
1594     /*
1595     * Serialize with zone creations.
1596     */
1597     mount_in_progress();
1598     /*
1599     * Instantiate (or reinstantiate) the file system. If appropriate,

```

```

1600     * splice it into the file system name space.
1601     *
1602     * We want VFS_MOUNT() to be able to override the vfs_resource
1603     * string if necessary (ie, mntfs), and also for a remount to
1604     * change the same (necessary when remounting '/' during boot).
1605     * So we set up vfs_mntpt and vfs_resource to what we think they
1606     * should be, then hand off control to VFS_MOUNT() which can
1607     * override this.
1608     *
1609     * For safety's sake, when changing vfs_resource or vfs_mntpt of
1610     * a vfs which is on the vfs list (i.e. during a remount), we must
1611     * never set those fields to NULL. Several bits of code make
1612     * assumptions that the fields are always valid.
1613     */
1614     vfs_swapopttbl(&mnt_mntopts, &vfsp->vfs_mntopts);
1615     if (remount) {
1616         if ((oldresource = vfsp->vfs_resource) != NULL)
1617             refstr_hold(oldresource);
1618         if ((oldmntpt = vfsp->vfs_mntpt) != NULL)
1619             refstr_hold(oldmntpt);
1620     }
1621     vfs_setresource(vfsp, resource, 0);
1622     vfs_setmntpoint(vfsp, mountpt, 0);
1623
1624     /*
1625     * going to mount on this vnode, so notify.
1626     */
1627     vnevent_mountedover(vp, NULL);
1628     error = VFS_MOUNT(vfsp, vp, uap, credp);
1629
1630     if (uap->flags & MS_RDONLY)
1631         vfs_setmntopt(vfsp, MNTOPT_RO, NULL, 0);
1632     if (uap->flags & MS_NOSUID)
1633         vfs_setmntopt(vfsp, MNTOPT_NOSUID, NULL, 0);
1634     if (uap->flags & MS_GLOBAL)
1635         vfs_setmntopt(vfsp, MNTOPT_GLOBAL, NULL, 0);
1636
1637     if (error) {
1638         lofi_remove(vfsp);
1639
1640         if (remount) {
1641             /* put back pre-remount options */
1642             vfs_swapopttbl(&mnt_mntopts, &vfsp->vfs_mntopts);
1643             vfs_setmntpoint(vfsp, refstr_value(oldmntpt),
1644                             VFSPP_VERBATIM);
1645             if (oldmntpt)
1646                 refstr_rele(oldmntpt);
1647             vfs_setresource(vfsp, refstr_value(oldresource),
1648                             VFSPP_VERBATIM);
1649             if (oldresource)
1650                 refstr_rele(oldresource);
1651             vfsp->vfs_flag = ovflags;
1652             vfs_unlock(vfsp);
1653             VFS_RELE(vfsp);
1654         } else {
1655             vfs_unlock(vfsp);
1656             vfs_freemnttab(vfsp);
1657             vfs_free(vfsp);
1658         }
1659     } else {
1660         /*
1661         * Set the mount time to now
1662         */
1663         vfsp->vfs_mtime = ddi_get_time();
1664         if (remount) {
1665             vfsp->vfs_flag &= ~VFS_REMOUNT;

```

```

1666             if (oldresource)
1667                 refstr_rele(oldresource);
1668             if (oldmntpt)
1669                 refstr_rele(oldmntpt);
1670         } else if (splice) {
1671             /*
1672             * Link vfsp into the name space at the mount
1673             * point. Vfs_add() is responsible for
1674             * holding the mount point which will be
1675             * released when vfs_remove() is called.
1676             */
1677             vfs_add(vp, vfsp, uap->flags);
1678         } else {
1679             /*
1680             * Hold the reference to file system which is
1681             * not linked into the name space.
1682             */
1683             vfsp->vfs_zone = NULL;
1684             VFS_HOLD(vfsp);
1685             vfsp->vfs_vnodecovered = NULL;
1686         }
1687     /*
1688     * Set flags for global options encountered
1689     */
1690     if (vfs_optionisset(vfsp, MNTOPT_RO, NULL))
1691         vfsp->vfs_flag |= VFS_RDONLY;
1692     else
1693         vfsp->vfs_flag &= ~VFS_RDONLY;
1694     if (vfs_optionisset(vfsp, MNTOPT_NOSUID, NULL)) {
1695         vfsp->vfs_flag |= (VFS_NOSETUID|VFS_NODEVICES);
1696     } else {
1697         if (vfs_optionisset(vfsp, MNTOPT_NODEVICES, NULL))
1698             vfsp->vfs_flag |= VFS_NODEVICES;
1699         else
1700             vfsp->vfs_flag &= ~VFS_NODEVICES;
1701         if (vfs_optionisset(vfsp, MNTOPT_NOSETUID, NULL))
1702             vfsp->vfs_flag |= VFS_NOSETUID;
1703         else
1704             vfsp->vfs_flag &= ~VFS_NOSETUID;
1705     }
1706     if (vfs_optionisset(vfsp, MNTOPT_NBMAND, NULL))
1707         vfsp->vfs_flag |= VFS_NBMAND;
1708     else
1709         vfsp->vfs_flag &= ~VFS_NBMAND;
1710
1711     if (vfs_optionisset(vfsp, MNTOPT_XATTR, NULL))
1712         vfsp->vfs_flag |= VFS_XATTR;
1713     else
1714         vfsp->vfs_flag &= ~VFS_XATTR;
1715
1716     if (vfs_optionisset(vfsp, MNTOPT_NOEXEC, NULL))
1717         vfsp->vfs_flag |= VFS_NOEXEC;
1718     else
1719         vfsp->vfs_flag &= ~VFS_NOEXEC;
1720
1721     /*
1722     * Now construct the output option string of options
1723     * we recognized.
1724     */
1725     if (uap->flags & MS_OPTIONSTR) {
1726         vfs_list_read_lock();
1727         copyout_error = vfs_buildoptionstr(
1728             &vfsp->vfs_mntopts, inargs, optlen);
1729         vfs_list_unlock();
1730         if (copyout_error == 0 &&
1731             (uap->flags & MS_SYSSPACE) == 0) {

```

```

1732         copyout_error = copyoutstr(inargs, opts,
1733         optlen, NULL);
1734     }
1735 }
1736
1737 /*
1738 * If this isn't a remount, set up the vopstats before
1739 * anyone can touch this. We only allow spliced file
1740 * systems (file systems which are in the namespace) to
1741 * have the VFS_STATS flag set.
1742 * NOTE: PxFs mounts the underlying file system with
1743 * MS_NOSPLICE set and copies those vfs_flags to its private
1744 * vfs structure. As a result, PxFs should never have
1745 * the VFS_STATS flag or else we might access the vfs
1746 * statistics-related fields prior to them being
1747 * properly initialized.
1748 */
1749 if (!remount && (vswp->vsw_flag & VSW_STATS) && splice) {
1750     initialize_vopstats(&vfsp->vfs_vopstats);
1751     /*
1752     * We need to set vfs_vskap to NULL because there's
1753     * a chance it won't be set below. This is checked
1754     * in teardown_vopstats() so we can't have garbage.
1755     */
1756     vfs->vfs_vskap = NULL;
1757     vfs->vfs_flag |= VFS_STATS;
1758     vfs->vfs_fstypevswp = get_fstype_vopstats(vfsp, vswp);
1759 }
1760
1761 if (vswp->vsw_flag & VSW_XID)
1762     vfs->vfs_flag |= VFS_XID;
1763
1764     vfs_unlock(vfsp);
1765 }
1766 mount_completed();
1767 if (splice)
1768     vn_vfsunlock(vp);
1769
1770 if ((error == 0) && (copyout_error == 0)) {
1771     if (!remount) {
1772         /*
1773         * Don't call get_vskstat_anchor() while holding
1774         * locks since it allocates memory and calls
1775         * VFS_STATVFS(). For NFS, the latter can generate
1776         * an over-the-wire call.
1777         */
1778         vskap = get_vskstat_anchor(vfsp);
1779         /* Only take the lock if we have something to do */
1780         if (vskap != NULL) {
1781             vfs_lock_wait(vfsp);
1782             if (vfs->vfs_flag & VFS_STATS) {
1783                 vfs->vfs_vskap = vskap;
1784             }
1785             vfs_unlock(vfsp);
1786         }
1787     }
1788     /* Return vfs to caller. */
1789     *vfsp = vfs;
1790     fsh_exec_mount_callbacks(vfsp);
1791 }
1792 errout:
1793     vfs_freeopttbl(&mnt_mntopts);
1794     if (resource != NULL)
1795         kmem_free(resource, strlen(resource) + 1);
1796     if (mountpt != NULL)
1797         kmem_free(mountpt, strlen(mountpt) + 1);

```

```

1798     /*
1799     * It is possible we errored prior to adding to mount in progress
1800     * table. Must free vnode we acquired with successful lookupname.
1801     */
1802     if (addmip)
1803         VN_RELE(bvp);
1804     if (delmip)
1805         vfs_delmip(vfsp);
1806     ASSERT(vswp != NULL);
1807     vfs_unrefvfsw(vswp);
1808     if (inargs != opts)
1809         kmem_free(inargs, MAX_MNTOPT_STR);
1810     if (copyout_error) {
1811         lofi_remove(vfsp);
1812         VFS_RELE(vfsp);
1813         error = copyout_error;
1814     }
1815     return (error);
1816 }
1817
1818 unchanged_portion_omitted
1819
1820 vfs_t EIO_vfs;
1821 vfsops_t *EIO_vfsops;
1822
1823 /*
1824 * Called from startup() to initialize all loaded vfs's
1825 */
1826 void
1827 vfsinit(void)
1828 {
1829     struct vfsw *vswp;
1830     int error;
1831     extern int vopstats_enabled;
1832     extern void vopstats_startup();
1833
1834     static const fs_operation_def_t EIO_vfsops_template[] = {
1835         VFSNAME_MOUNT, { .error = vfs_EIO },
1836         VFSNAME_UNMOUNT, { .error = vfs_EIO },
1837         VFSNAME_ROOT, { .error = vfs_EIO },
1838         VFSNAME_STATVFS, { .error = vfs_EIO },
1839         VFSNAME_SYNC, { .vfs_sync = vfs_EIO_sync },
1840         VFSNAME_VGET, { .error = vfs_EIO },
1841         VFSNAME_MOUNTROOT, { .error = vfs_EIO },
1842         VFSNAME_FREEVFS, { .error = vfs_EIO },
1843         VFSNAME_VNSTATE, { .error = vfs_EIO },
1844         NULL, NULL
1845     };
1846
1847     static const fs_operation_def_t stray_vfsops_template[] = {
1848         VFSNAME_MOUNT, { .error = vfststray },
1849         VFSNAME_UNMOUNT, { .error = vfststray },
1850         VFSNAME_ROOT, { .error = vfststray },
1851         VFSNAME_STATVFS, { .error = vfststray },
1852         VFSNAME_SYNC, { .vfs_sync = vfststray_sync },
1853         VFSNAME_VGET, { .error = vfststray },
1854         VFSNAME_MOUNTROOT, { .error = vfststray },
1855         VFSNAME_FREEVFS, { .error = vfststray },
1856         VFSNAME_VNSTATE, { .error = vfststray },
1857         NULL, NULL
1858     };
1859
1860     /* Create vfs cache */
1861     vfs_cache = kmem_cache_create("vfs_cache", sizeof (struct vfs),
1862         sizeof (uintptr_t), NULL, NULL, NULL, NULL, 0);
1863
1864     /* Initialize the vnode cache (file systems may use it during init). */

```



```

4236     vn_create_cache();
4238     /* Setup event monitor framework */
4239     fem_init();
4241     /* Setup filesystem hook framework */
4242     fsh_init();
4244     /* Initialize the dummy stray file system type. */
4245     error = vfs_setfsops(0, stray_vfsops_template, NULL);
4247     /* Initialize the dummy EIO file system. */
4248     error = vfs_makefsops(EIO_vfsops_template, &EIO_vfsops);
4249     if (error != 0) {
4250         cmn_err(CE_WARN, "vfsinit: bad EIO vfs ops template");
4251         /* Shouldn't happen, but not bad enough to panic */
4252     }
4254     VFS_INIT(&EIO_vfs, EIO_vfsops, (caddr_t)NULL);
4256     /*
4257     * Default EIO_vfs.vfs_flag to VFS_UNMOUNTED so a lookup
4258     * on this vfs can immediately notice it's invalid.
4259     */
4260     EIO_vfs.vfs_flag |= VFS_UNMOUNTED;
4262     /*
4263     * Call the init routines of non-loadable filesystems only.
4264     * Filesystems which are loaded as separate modules will be
4265     * initialized by the module loading code instead.
4266     */
4268     for (vswp = &vfssw[1]; vswp < &vfssw[infstypel]; vswp++) {
4269         RLOCK_VFSSW();
4270         if (vswp->vsw_init != NULL)
4271             (*vswp->vsw_init)(vswp - vfssw, vswp->vsw_name);
4272         RUNLOCK_VFSSW();
4273     }
4275     vopstats_startup();
4277     if (vopstats_enabled) {
4278         /* EIO_vfs can collect stats, but we don't retrieve them */
4279         initialize_vopstats(&EIO_vfs.vfs_vopstats);
4280         EIO_vfs.vfs_fstypevsp = NULL;
4281         EIO_vfs.vfs_vskap = NULL;
4282         EIO_vfs.vfs_flag |= VFS_STATS;
4283     }
4285     xattr_init();
4287     reparse_point_init();
4288 }
    unchanged portion omitted
4305 void
4306 vfs_free(vfs_t *vfsp)
4307 {
4308     /*
4309     * One would be tempted to assert that "vfsp->vfs_count == 0".
4310     * The problem is that this gets called out of domount() with
4311     * a partially initialized vfs and a vfs_count of 1. This is
4312     * also called from vfs_rele() with a vfs_count of 0. We can't
4313     * call VFS_RELE() from domount() if VFS_MOUNT() hasn't successfully
4314     * returned. This is because VFS_MOUNT() fully initializes the
4315     * vfs structure and its associated data. VFS_RELE() will call

```

```

4316     * VFS_FREEVFS() which may panic the system if the data structures
4317     * aren't fully initialized from a successful VFS_MOUNT().
4318     */
4320     /* If FEM was in use, make sure everything gets cleaned up */
4321     if (vfsp->vfs_femhead) {
4322         ASSERT(vfsp->vfs_femhead->femh_list == NULL);
4323         mutex_destroy(&vfsp->vfs_femhead->femh_lock);
4324         kmem_free(vfsp->vfs_femhead, sizeof (*vfsp->vfs_femhead));
4325         vfsp->vfs_femhead = NULL;
4326     }
4328     /*
4329     * fsh cleanup
4330     * There's no need here to use atomic operations on vfs_fshrecord.
4331     */
4332     if (vfsp->vfs_fshrecord != NULL) {
4333         fsh_fsrec_destroy(vfsp->vfs_fshrecord);
4334         vfsp->vfs_fshrecord = NULL;
4335     }
4337     if (vfsp->vfs_implp)
4338         vfsimpl_tearardown(vfsp);
4339     sema_destroy(&vfsp->vfs_reflock);
4340     kmem_cache_free(vfs_cache, vfsp);
4341 }
    unchanged portion omitted
4353 /*
4354 * Decrements the vfs reference count by one atomically. When
4355 * vfs reference count becomes zero, it calls the file system
4356 * specific vfs_freevfs() to free up the resources.
4357 */
4358 void
4359 vfs_rele(vfs_t *vfsp)
4360 {
4361     ASSERT(vfsp->vfs_count != 0);
4362     if (atomic_add_32_nv(&vfsp->vfs_count, -1) == 0) {
4363         fsh_exec_free_callbacks(vfsp);
4364         VFS_FREEVFS(vfsp);
4365         lofi_remove(vfsp);
4366         if (vfsp->vfs_zone)
4367             zone_rele_ref(&vfsp->vfs_implp->vi_zone_ref,
4368                 ZONE_REF_VFS);
4369         vfs_freemttab(vfsp);
4370         vfs_free(vfsp);
4371     }
4372 }
    unchanged portion omitted

```

```

*****
105122 Tue Sep 3 18:09:41 2013
new/usr/src/uts/common/fs/vnode.c
fsh, fsd, libfsd, fsdam
*****
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 (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */

22 /*
23  * Copyright (c) 1988, 2010, Oracle and/or its affiliates. All rights reserved.
24 */

26 /*      Copyright (c) 1983, 1984, 1985, 1986, 1987, 1988, 1989 AT&T      */
27 /*      All Rights Reserved      */

29 /*
30  * University Copyright- Copyright (c) 1982, 1986, 1988
31  * The Regents of the University of California
32  * All Rights Reserved
33  *
34  * University Acknowledgment- Portions of this document are derived from
35  * software developed by the University of California, Berkeley, and its
36  * contributors.
37  */

39 #include <sys/types.h>
40 #include <sys/param.h>
41 #include <sys/t_lock.h>
42 #include <sys/errno.h>
43 #include <sys/cred.h>
44 #include <sys/user.h>
45 #include <sys/uio.h>
46 #include <sys/file.h>
47 #include <sys/pathname.h>
48 #include <sys/vfs.h>
49 #include <sys/vfs_opreg.h>
50 #include <sys/vnode.h>
51 #include <sys/rwstlock.h>
52 #include <sys/fem.h>
53 #include <sys/stat.h>
54 #include <sys/mode.h>
55 #include <sys/conf.h>
56 #include <sys/sysmacros.h>
57 #include <sys/cmn_err.h>
58 #include <sys/system.h>
59 #include <sys/kmem.h>
60 #include <sys/debug.h>
61 #include <c2/audit.h>

```

```

62 #include <sys/acl.h>
63 #include <sys/nbmlck.h>
64 #include <sys/fcntl.h>
65 #include <fs/fs_subr.h>
66 #include <sys/taskq.h>
67 #include <fs/fs_reparse.h>
68 #include <sys/fsh_impl.h>

70 /* Determine if this vnode is a file that is read-only */
71 #define ISROFILE(vp) \
72     ((vp)->v_type != VCHR && (vp)->v_type != VBLK && \
73      (vp)->v_type != VFIFO && vn_is_readonly(vp))

75 /* Tunable via /etc/system; used only by admin/install */
76 int nfs_global_client_only;

78 /*
79  * Array of vopstats_t for per-FS-type vopstats. This array has the same
80  * number of entries as and parallel to the vfssw table. (Arguably, it could
81  * be part of the vfssw table.) Once it's initialized, it's accessed using
82  * the same fstype index that is used to index into the vfssw table.
83  */
84 vopstats_t **vopstats_fstype;

86 /* vopstats initialization template used for fast initialization via bcopy() */
87 static vopstats_t *vs_templatep;

89 /* Kmem cache handle for vsk_anchor_t allocations */
90 kmem_cache_t *vsk_anchor_cache;

92 /* file events cleanup routine */
93 extern void free_fopdata(vnode_t *);

95 /*
96  * Root of AVL tree for the kstats associated with vopstats. Lock protects
97  * updates to vsktat_tree.
98  */
99 avl_tree_t      vskstat_tree;
100 kmutex_t        vskstat_tree_lock;

102 /* Global variable which enables/disables the vopstats collection */
103 int vopstats_enabled = 1;

105 /*
106  * forward declarations for internal vnode specific data (vsd)
107  */
108 static void *vsd_realloc(void *, size_t, size_t);

110 /*
111  * forward declarations for reparse point functions
112  */
113 static int fs_reparse_mark(char *target, vattn_t *vap, xvattr_t *xvattr);

115 /*
116  * VSD -- VNODE SPECIFIC DATA
117  * The v_data pointer is typically used by a file system to store a
118  * pointer to the file system's private node (e.g. ufs inode, nfs rnode).
119  * However, there are times when additional project private data needs
120  * to be stored separately from the data (node) pointed to by v_data.
121  * This additional data could be stored by the file system itself or
122  * by a completely different kernel entity. VSD provides a way for
123  * callers to obtain a key and store a pointer to private data associated
124  * with a vnode.
125  *
126  * Callers are responsible for protecting the vsd by holding v_vsd_lock
127  * for calls to vsd_set() and vsd_get().

```

```

128 */
130 /*
131  * vsd_lock protects:
132  *   vsd_nkeys - creation and deletion of vsd keys
133  *   vsd_list - insertion and deletion of vsd_node in the vsd_list
134  *   vsd_destructor - adding and removing destructors to the list
135  */
136 static kmutex_t      vsd_lock;
137 static uint_t        vsd_nkeys;      /* size of destructor array */
138 /* list of vsd_node's */
139 static list_t *vsd_list = NULL;
140 /* per-key destructor funcs */
141 static void          (**vsd_destructor)(void *);

143 /*
144  * The following is the common set of actions needed to update the
145  * vopstats structure from a vnode op.  Both VOPSTATS_UPDATE() and
146  * VOPSTATS_UPDATE_IO() do almost the same thing, except for the
147  * recording of the bytes transferred.  Since the code is similar
148  * but small, it is nearly a duplicate.  Consequently any changes
149  * to one may need to be reflected in the other.
150  * Rundown of the variables:
151  * vp - Pointer to the vnode
152  * counter - Partial name structure member to update in vopstats for counts
153  * bytecounter - Partial name structure member to update in vopstats for bytes
154  * bytesval - Value to update in vopstats for bytes
155  * fstype - Index into vsanchor_fstype[], same as index into vfssw[]
156  * vsp - Pointer to vopstats structure (either in vfs or vsanchor_fstype[i])
157  */

159 #define VOPSTATS_UPDATE(vp, counter) {
160     vfs_t *vfsp = (vp)->v_vfsp;
161     if (vfsp && vfsp->vfs_implp &&
162         (vfsp->vfs_flag & VFS_STATS) && (vp)->v_type != VBAD) {
163         vopstats_t *vsp = &vfsp->vfs_vopstats;
164         uint64_t *stataddr = &(vsp->n##counter.value.ui64);
165         extern void __dtrace_probe__fsinfo_##counter(vnode_t *,
166             size_t, uint64_t *);
167         __dtrace_probe__fsinfo_##counter(vp, 0, stataddr);
168         (*stataddr)++;
169         if ((vsp = vfsp->vfs_fstypevsp) != NULL) {
170             vsp->n##counter.value.ui64++;
171         }
172     }
173 }

```

unchanged portion omitted

```

3216 int
3217 fop_read(
3218     vnode_t *vp,
3219     uio_t *uiop,
3220     int ioflag,
3221     cred_t *cr,
3222     caller_context_t *ct)
3223 {
3224     int err;
3225     ssize_t resid_start = uiop->uio_resid;

3227     VOPXID_MAP_CR(vp, cr);

3229     err = fsh_read(vp, uiop, ioflag, cr, ct);
3228     err = (*(vp)->v_op->vop_read)(vp, uiop, ioflag, cr, ct);
3230     VOPSTATS_UPDATE_IO(vp, read,
3231         read_bytes, (resid_start - uiop->uio_resid));
3232     return (err);

```

```

3233 }

3235 int
3236 fop_write(
3237     vnode_t *vp,
3238     uio_t *uiop,
3239     int ioflag,
3240     cred_t *cr,
3241     caller_context_t *ct)
3242 {
3243     int err;
3244     ssize_t resid_start = uiop->uio_resid;

3246     VOPXID_MAP_CR(vp, cr);

3248     err = fsh_write(vp, uiop, ioflag, cr, ct);
3247     err = (*(vp)->v_op->vop_write)(vp, uiop, ioflag, cr, ct);
3249     VOPSTATS_UPDATE_IO(vp, write,
3250         write_bytes, (resid_start - uiop->uio_resid));
3251     return (err);
3252 }

```

unchanged portion omitted

```

*****
22805 Tue Sep 3 18:09:41 2013
new/usr/src/uts/common/io/fsd/fsd.c
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel. All rights reserved.
14 */

16 /*
17  * Filesystem disturber pseudo-device driver.
18 */

20 #include <sys/conf.h>
21 #include <sys/ddi.h>
22 #include <sys/file.h>
23 #include <sys/fsd.h>
24 #include <sys/fsh.h>
25 #include <sys/kmem.h>
26 #include <sys/ksynch.h>
27 #include <sys/list.h>
28 #include <sys/mkdev.h>
29 #include <sys/refstr.h>
30 #include <sys/stat.h>
31 #include <sys/sunddi.h>
32 #include <sys/sysmacros.h>
33 #include <sys/types.h>

35 /*
36  * TODO:
37  * - add checking if a file descriptor passed by the client is indeed
38  *   a mountpoint (we'd like to avoid disturbing / instead of an
39  *   unmounted filesystem)
40 */
41 /*
42  * fsd - filesystem disturber
43  *
44  * 1. Abstract
45  * Filesystem disturber is a pseudo-device driver used to inject pathological
46  * behaviour into vfs calls. It is NOT a fuzzer. That kind of behaviour
47  * should be expected and correctly handled by software. A simple example of
48  * such behaviour is read() reading less bytes than it was requested. It's
49  * well documented and every read() caller should check the return value of
50  * this function before proceeding.
51  *
52  * 2. Features
53  * * per-vfs injections
54  * * injection installing on every newly mounted vfs (that's called an
55  *   omnipresent disturber)
56  *
57  * 3. Usage
58  * fsd_t is a structure which contains all the parameters for the disturbers.
59  * This structure is shared by all hooks on a vfs_t.
60  *
61  * fsd_info_t is filled out by a call to ioctl() and it provides basic

```

```

62  * information about fsd's current status.
63  *
64  * fsd_dis_t is passed to ioctl() when a request to disturb a filesystem is
65  * made. It's just a descriptor of a representative file and an fsd_t structure.
66  *
67  * fsd_fs_t is a structure filled out by ioctl() call when the client requests a
68  * full list of disturbers installed in the system.
69  *
70  * fsd_ioc_t is an union for different ioctl() commands.
71  *
72  * ioctl() commands:
73  * FSD_ENABLE:
74  *   ioctl(fd, FSD_ENABLE);
75  *   Enables the fsd. When fsd is enabled, any attempts to detach the driver
76  *   will fail.
77  *
78  * FSD_DISABLE:
79  *   ioctl(fd, FSD_DISABLE);
80  *   Disables the fsd.
81  *
82  * FSD_GET_PARAM:
83  *   ioctl(fd, FSD_GET_PARAM, ioc);
84  *   Get's fsd_t associated with a given filesystem. ioc is fsdioc_mnt when
85  *   passed to ioctl(). fsdioc_param is the output.
86  *   Errors:
87  *       ENOENT - the filesystem is not being disturbed
88  *
89  * FSD_DISTURB:
90  *   ioctl(fd, FSD_DISTURB, ioc);
91  *   Installs a disturber on a given filesystem. If a disturber is already
92  *   installed on this filesystem, it overwrites it. ioc is fsdioc_dis.
93  *   Errors:
94  *       EAGAIN - hook limit exceeded
95  *       EBADF - cannot open the file descriptor
96  *       EINVAL - parameters are invalid
97  *
98  * FSD_DISTURB_OFF:
99  *   ioctl(fd, FSD_DISTURB_OFF, ioc);
100  *   Removes a disturber from a given filesystem. ioc is fsdioc_mnt
101  *   Errors:
102  *       EBADF - cannot open the file descriptor
103  *       ENOENT - the filesystem is not being disturbed
104  *
105  * FSD_DISTURB_OMNI:
106  *   ioctl(fd, FSD_DISTURB_OMNI, ioc);
107  *   Install an omnipresent disturber. It means that whenever a new vfs_t is
108  *   being created, this disturber is installed on it. If an omnipresent
109  *   disturber is already installed, it overwrites it. ioc is fsdioc_param
110  *   Errors:
111  *       EINVAL - parameters are invalid
112  *
113  * FSD_DISTURB_OMNI_OFF:
114  *   ioctl(fd, FSD_DISTURB_OMNI_OFF);
115  *   Removes the omnipresent disturber. That does NOT mean that filesystems
116  *   which are disturbed because of the omnipresent disturber presence in the
117  *   past are going to stop being disturbed after this call.
118  *
119  * FSD_GET_LIST:
120  *   ioctl(fd, FSD_GET_LIST, ioc);
121  *   Get's a full list of disturbers installed in the system. ioc is
122  *   fsdioc_list here. This is a structure with two fields, count and listp.
123  *   The count is the number of fsd_fs_t's allocated on the address that
124  *   listp is pointing to. There would be at most count fsd_fs_t entries
125  *   copied out to the caller. Also, count is set to the number of entries
126  *   copied out.
127  *

```

```

128 * FSD_GET_INFO:
129 *   ioctl(fd, FSD_GET_INFO, ioc);
130 *   Get's current information about fsd. ioc is fsdioc_info here.
131 *
132 * At most one hook is installed per vfs_t, and fsd_t describes all possible
133 * disturbance methods. Multiple commands using the fsd should somehow cooperate
134 * in order not to destroy each other efforts in installing disturbers.
135 *
136 * 4. Internals
137 * When fsd_enabled is nonzero, fsd_detach() fails.
138 *
139 * These mount callback is used for installing injections on newly mounted
140 * vfs_t's (omnipresent).
141 *
142 * The list of currently installed hooks is kept in fsd_list.
143 *
144 * fsd installs at most one hook on a vfs_t.
145 *
146 * Inside fsd_detach, we go through fsd_hooks list. There is no guarantee that
147 * a hook remove callback (fsd_remove_cb) wouldn't execute inside
148 * fsh_hook_remove(), thus we can't assume that while walking through fsd_hooks,
149 * our iterator will be valid, because fsh_hook_remove() could invalidate it.
150 * That's why fsd_detaching flag is introduced.
151 *
152 * 5. Locking
153 * Every modification of fsd_enable, fsd_hooks, fsd_omni_param and fsd_list is
154 * protected by fsd_lock.
155 *
156 * Hooks use only the elements of fsd_list, nothing else. Before an element of
157 * fsd_list is destroyed, a hook which uses it is removed. Elements from
158 * fsd_lists are removed and destroyed in the hook remove callback
159 * (fsd_remove_cb).
160 *
161 * Because of the fact that fsd_remove_cb() could be called both in the context
162 * of the thread that executes fsh_hook_remove() or outside the fsd, we need to
163 * use fsd_rem_thread in order not to cause a deadlock. fsh_hook_remove() could
164 * be called by at most one thread inside fsd (fsd_remove_disturber() holds
165 * fsd_lock). We just have to check inside fsd_remove_cb() if it was called
166 * from fsh_hook_remove() or not. We use fsd_rem_thread to determine that.
167 *
168 * fsd_int_t.fsd_i_param is protected by fsd_int_t.fsd_i_lock which is an rwlock.
169 */
171 /*
172 * Once a set of hooks is installed on a filesystem, there's no need
173 * to bother fsh if we want to change the parameters of disturbance.
174 * Instead, we use fsd_lock to protect the fsd_int_t when it's being
175 * used or changed.
176 */
177 typedef struct fsd_int {
178     krwlock_t    fsdi_lock;    /* protects fsd_param */
179     fsd_t        fsdi_param;
180     fsh_handle_t fsdi_handle;  /* we use fsh's handle in fsd */
181     vfs_t        fsdi_vfsp;
182     int          fsdi_doomed;
183     list_node_t  fsdi_next;
184 } fsd_int_t;
186 static dev_info_t *fsd_devi;
188 /*
189 * fsd_lock protects: fsd_enabled, fsd_omni_param, fsd_list, fsd_cb_handle,
190 * fsd_detaching
191 */
192 /*
193 static kmutex_t fsd_lock;

```

```

195 static kthread_t *fsd_rem_thread;
196 static kmutex_t fsd_rem_thread_lock;
198 static fsd_t *fsd_omni_param; /* Argument used by fsd's mount callback. */
199 static fsh_callback_handle_t fsd_cb_handle;
200 static int fsd_enabled;
201 static int fsd_detaching;
203 /*
204 * List of fsd_int_t. For every vfs_t on which fsd has installed a set of hooks
205 * there exist exactly one fsd_int_t with fsdi_vfsp pointing to this vfs_t.
206 */
207 static list_t fsd_list;
208 static int fsd_list_count;
209 static kcondvar_t fsd_cv_empty;
212 /*
213 * Although it's safe to use this kind of pseudo-random number generator,
214 * it behaves very regular when it comes to parity. Every fsd_rand() call
215 * changes the parity of the seed. That's why when a range of width 2 is set
216 * as a parameter, it's highly possible that the random value will always be
217 * the same, because fsd_rand() could be called the same number of times in a
218 * hook.
219 */
220 static long    fsd_rand_seed;
222 static int
223 fsd_rand()
224 {
225     fsd_rand_seed = fsd_rand_seed * 1103515245L + 12345;
226     return (fsd_rand_seed & 0x7fffffff);
227 }
229 /* vnode hooks */
230 /*
231 * A pointer to a given fsd_int_t is valid always inside fsh_hook_xxx()
232 * call, because it's valid until the hooks associated with it are removed.
233 * If a hook is removed, it cannot be executing.
234 */
235 static int
236 fsd_hook_read(fsh_int_t *fshi, void *arg, vnode_t *vp, uio_t *uiop,
237             int ioflag, cred_t *cr, caller_context_t *ct)
238 {
239     fsd_int_t *fsdi = (fsd_int_t *)arg;
240     uint64_t count, less, less_chance;
242     /*
243     * It is used to keep an odd number of fsd_rand() calls in every
244     * fsd_hook_read() call. That is desired because when a range of width
245     * 2 is set as a parameter, we don't want to make it a constant.
246     * The pseudo-random number generator returns a number with different
247     * parity with every call. If this function is called in every
248     * fsd_hook_read() execution even number of times, it would always be
249     * the same % 2.
250     */
251     (void) fsd_rand();
253     ASSERT(vp->v_vfsp == fsdi->fsdi_vfsp);
255     rw_enter(&fsdi->fsdi_lock, RW_READER);
256     less_chance = fsdi->fsdi_param.read_less_chance;
257     less = (uint64_t)fsd_rand() %
258           (fsdi->fsdi_param.read_less_r[1] + 1 -
259            fsdi->fsdi_param.read_less_r[0]) + fsdi->fsdi_param.read_less_r[0];

```

```

260     rw_exit(&fsdi->fsdi_lock);

262     count = uiop->uio_iov->iiov_len;
263     if ((uint64_t)fsd_rand() % 100 < less_chance) {
264         extern size_t copyout_max_cached;
265         int ret;

267         if (count > less)
268             count -= less;
269         else
270             less = 0;

272         uiop->uio_iov->iiov_len = count;
273         uiop->uio_resid = count;
274         if (count <= copyout_max_cached)
275             uiop->uio_extflg = UIO_COPY_CACHED;
276         else
277             uiop->uio_extflg = UIO_COPY_DEFAULT;

279         ret = fsh_next_read(fshi, vp, uiop, ioflag, cr, ct);
280         uiop->uio_resid += less;
281         return (ret);
282     }

284     return (fsh_next_read(fshi, vp, uiop, ioflag, cr, ct));
285 }

288 static void
289 fsd_remove_cb(void *arg, fsh_handle_t handle)
290 {
291     _NOTE(ARGUNUSED(handle));

293     fsd_int_t *fsdi = (fsd_int_t *)arg;
294     int fsd_context;

296     mutex_enter(&fsd_rem_thread_lock);
297     fsd_context = fsd_rem_thread == curthread;
298     mutex_exit(&fsd_rem_thread_lock);

300     if (!fsd_context)
301         mutex_enter(&fsd_lock);

303     ASSERT(MUTEX_HELD(&fsd_lock));

305     if (!fsd_detaching)
306         list_remove(&fsd_list, fsdi);

308     rw_destroy(&fsdi->fsdi_lock);
309     kmem_free(fsdi, sizeof (*fsdi));

311     fsd_list_count--;
312     if (fsd_list_count == 0)
313         cv_signal(&fsd_cv_empty);

315     if (!fsd_context)
316         mutex_exit(&fsd_lock);
317 }

319 /*
320  * Installs a set of hook with given parameters on a vfs_t.
321  *
322  * It is expected that fsd_lock is being held.
323  *
324  * Returns 0 on success and non-zero if hook limit exceeded.
325  */

```

```

326 static int
327 fsd_install_disturber(vfs_t *vfsp, fsd_t *fsd)
328 {
329     fsd_int_t *fsdi;

331     ASSERT(MUTEX_HELD(&fsd_lock));

333     for (fsdi = list_head(&fsd_list); fsdi != NULL;
334          fsdi = list_next(&fsd_list, fsdi)) {
335         if (fsdi->fsdi_vfsp == vfsp)
336             break;
337     }

339     if (fsdi != NULL) {
340         /* Just change the existing fsd_int_t */
341         rw_enter(&fsdi->fsdi_lock, RW_WRITER);
342         (void) memcpy(&fsdi->fsdi_param, fsd,
343                     sizeof (fsdi->fsdi_param));
344         rw_exit(&fsdi->fsdi_lock);
345     } else {
346         fsh_t hook = { 0 };

348         fsdi = kmem_zalloc(sizeof (*fsdi), KM_SLEEP);
349         fsdi->fsdi_vfsp = vfsp;
350         (void) memcpy(&fsdi->fsdi_param, fsd,
351                     sizeof (fsdi->fsdi_param));
352         rw_init(&fsdi->fsdi_lock, NULL, RW_DRIVER, NULL);

354         hook.arg = fsdi;
355         hook.read = fsd_hook_read;
356         hook.remove_cb = fsd_remove_cb;

358         /*
359          * It is safe to do so, because none of the hooks installed
360          * by fsd uses fsdi_handle nor the fsd_list.
361          */
362         fsdi->fsdi_handle = fsh_hook_install(vfsp, &hook);
363         if (fsdi->fsdi_handle == -1) {
364             kmem_free(fsdi, sizeof (*fsdi));
365             rw_destroy(&fsdi->fsdi_lock);
366             return (-1);
367         }
368         list_insert_head(&fsd_list, fsdi);
369         fsd_list_count++;
370     }
371     return (0);
372 }

374 static int
375 fsd_remove_disturber(vfs_t *vfsp)
376 {
377     fsd_int_t *fsdi;

379     ASSERT(MUTEX_HELD(&fsd_lock));

381     for (fsdi = list_head(&fsd_list); fsdi != NULL;
382          fsdi = list_next(&fsd_list, fsdi)) {
383         if (fsdi->fsdi_vfsp == vfsp)
384             break;
385     }
386     if (fsdi == NULL || fsdi->fsdi_doomed)
387         return (ENOENT);

389     fsdi->fsdi_doomed = 1;

391     mutex_enter(&fsd_rem_thread_lock);

```

```

392 fsd_rem_thread = curthread;
393 mutex_exit(&fsd_rem_thread_lock);

395 ASSERT(fsh_hook_remove(fsdi->fsdi_handle) == 0);

397 mutex_enter(&fsd_rem_thread_lock);
398 fsd_rem_thread = NULL;
399 mutex_exit(&fsd_rem_thread_lock);

401 return (0);
402 }

404 static void
405 fsd_callback_mount(vfs_t *vfsp, void *arg)
406 {
407     _NOTE(ARGUNUSED(arg));

409     int error = 0;

411     mutex_enter(&fsd_lock);
412     if (fsd_omni_param != NULL)
413         error = fsd_install_disturber(vfsp, fsd_omni_param);
414     mutex_exit(&fsd_lock);

416     if (error != 0) {
417         refstr_t *mntref;

419         mntref = vfs_getmntpoint(vfsp);
420         (void) cmn_err(CE_NOTE, "Installing disturber for %s failed.\n",
421             refstr_value(mntref));
422         refstr_rele(mntref);
423     }
424 }

426 static void
427 fsd_enable()
428 {
429     mutex_enter(&fsd_lock);
430     fsd_enabled = 1;
431     mutex_exit(&fsd_lock);
432 }

434 static void
435 fsd_disable()
436 {
437     mutex_enter(&fsd_lock);
438     fsd_enabled = 0;
439     mutex_exit(&fsd_lock);
440 }

443 /* Entry points */
444 static int
445 fsd_attach(dev_info_t *dip, ddi_attach_cmd_t cmd)
446 {
447     minor_t instance;
448     fsh_callback_t cb = { 0 };

450     if (cmd != DDI_ATTACH)
451         return (DDI_FAILURE);

453     if (fsd_devi != NULL)
454         return (DDI_FAILURE);

456     instance = ddi_get_instance(dip);
457     if (ddi_create_minor_node(dip, "fsd", S_IFCHR, instance,

```

```

458     DDI_PSEUDO, 0) == DDI_FAILURE)
459         return (DDI_FAILURE);
460     fsd_devi = dip;
461     ddi_report_dev(fsd_devi);

463     list_create(&fsd_list, sizeof (fsd_int_t),
464         offsetof(fsd_int_t, fsdi_next));

466     fsd_rand_seed = gethrtime();

468     mutex_init(&fsd_lock, NULL, MUTEX_DRIVER, NULL);
469     mutex_init(&fsd_rem_thread_lock, NULL, MUTEX_DRIVER, NULL);
470     cv_init(&fsd_cv_empty, NULL, CV_DRIVER, NULL);

472     cb.fshc_mount = fsd_callback_mount;
473     cb.fshc_arg = fsd_omni_param;
474     fsd_cb_handle = fsh_callback_install(&cb);
475     if (fsd_cb_handle == -1) {
476         /* Cleanup */
477         list_destroy(&fsd_list);
478         cv_destroy(&fsd_cv_empty);
479         mutex_destroy(&fsd_rem_thread_lock);
480         mutex_destroy(&fsd_lock);
481         ddi_remove_minor_node(fsd_devi, NULL);
482         fsd_devi = NULL;
483         return (DDI_FAILURE);
484     }

486     return (DDI_SUCCESS);
487 }

489 /*
490  * If fsd_enable() was called and there was no subsequent fsd_disable() call,
491  * detach will fail.
492  */
493 static int
494 fsd_detach(dev_info_t *dip, ddi_detach_cmd_t cmd)
495 {
496     fsd_int_t *fsdi;

498     if (cmd != DDI_DETACH)
499         return (DDI_FAILURE);

501     ASSERT(dip == fsd_devi);

503     /*
504      * No need to hold fsd_lock here. Since only the hooks and callbacks
505      * might be running at this point.
506      */
507     if (fsd_enabled)
508         return (DDI_FAILURE);

510     ddi_remove_minor_node(dip, NULL);
511     fsd_devi = NULL;

513     mutex_enter(&fsd_lock);
514     fsd_detaching = 1;
515     while ((fsdi = list_remove_head(&fsd_list)) != NULL)
516         if (fsdi->fsdi_doomed == 0) {
517             fsdi->fsdi_doomed = 1;

519             mutex_enter(&fsd_rem_thread_lock);
520             fsd_rem_thread = curthread;
521             mutex_exit(&fsd_rem_thread_lock);
522         }
523     /*

```

```

524         * fsd_lock is held, so no other thread could have
525         * removed this hook.
526         */
527         ASSERT(fsh_hook_remove(fsdi->fsdi_handle) == 0);

529         mutex_enter(&fsd_rem_thread_lock);
530         fsd_rem_thread = NULL;
531         mutex_exit(&fsd_rem_thread_lock);
532     }

534     while (fsd_list_count > 0)
535         cv_wait(&fsd_cv_empty, &fsd_lock);
536     mutex_exit(&fsd_lock);
537     cv_destroy(&fsd_cv_empty);

539     ASSERT(fsh_callback_remove(fsd_cb_handle) == 0);
540     if (fsd_omni_param != NULL) {
541         kmem_free(fsd_omni_param, sizeof (*fsd_omni_param));
542         fsd_omni_param = NULL;
543     }

545     /* After removing the callback and hooks, it is safe to remove these */
546     list_destroy(&fsd_list);
547     mutex_destroy(&fsd_rem_thread_lock);
548     mutex_destroy(&fsd_lock);

550     return (DDI_SUCCESS);
551 }

553 static int
554 fsd_getinfo(dev_info_t *dip, ddi_info_cmd_t infocmd, void *arg, void **resultp)
555 {
556     _NOTE(ARGUNUSED(dip));

558     switch (infocmd) {
559     case DDI_INFO_DEVT2DEVINFO:
560         *resultp = fsd_dev;
561         return (DDI_SUCCESS);
562     case DDI_INFO_DEVT2INSTANCE:
563         *resultp = (void *) (uintptr_t) getminor((dev_t) arg);
564         return (DDI_SUCCESS);
565     default:
566         return (DDI_FAILURE);
567     }
568 }

570 static int
571 fsd_open(dev_t *devp, int flag, int otyp, cred_t *credp)
572 {
573     _NOTE(ARGUNUSED(devp));

575     if (flag & FXCL || flag & FNDELAY)
576         return (EINVAL);

578     if (otyp != OTYP_CHR)
579         return (EINVAL);

581     if (!(flag & FREAD && flag & FWRITE))
582         return (EINVAL);

584     if (drv_priv(credp) == EPERM)
585         return (EPERM);

587     return (0);
588 }

```

```

590 static int
591 fsd_close(dev_t dev, int flag, int otyp, cred_t *credp)
592 {
593     _NOTE(ARGUNUSED(dev));
594     _NOTE(ARGUNUSED(flag));
595     _NOTE(ARGUNUSED(otyp));
596     _NOTE(ARGUNUSED(credp));

598     return (0);
599 }

602 /* ioctl(9E) and it's support functions */
603 static int
604 fsd_check_param(fsd_t *fsd)
605 {
606     if (fsd->read_less_chance > 100 ||
607         fsd->read_less_r[0] > fsd->read_less_r[1])
608         return (EINVAL);
609     return (0);
610 }

612 static int
613 fsd_ioctl_disturb(fsd_ioc_t *ioc, int mode, int *rvalp)
614 {
615     file_t *file;
616     fsd_dis_t dis;
617     int rv;

619     if (ddi_copyin(&ioc->fsdioc_dis, &dis, sizeof (dis), mode))
620         return (EFAULT);

622     if ((rv = fsd_check_param(&dis.fsdd_param)) != 0) {
623         *rvalp = rv;
624         return (0);
625     }

627     if ((file = getf((int)dis.fsdd_mnt)) == NULL) {
628         *rvalp = EBADF;
629         return (0);
630     }

632     mutex_enter(&fsd_lock);
633     rv = fsd_install_disturber(file->v_vnode->v_vfsp, &dis.fsdd_param);
634     mutex_exit(&fsd_lock);

636     releasef((int)dis.fsdd_mnt);

638     if (rv != 0)
639         *rvalp = EAGAIN;
640     else
641         *rvalp = 0;

643     return (0);
644 }

646 static int
647 fsd_ioctl_get_param(fsd_ioc_t *ioc, int mode, int *rvalp)
648 {
649     file_t *file;
650     fsd_int_t *fsdi;
651     int error = 0;
652     int64_t fd;
653     vfs_t *vfsp;

655     if (ddi_copyin(&ioc->fsdioc_mnt, &fd, sizeof (fd), mode))

```



```

656         return (EFAULT);

658     if ((file = getf((int)fd)) == NULL) {
659         *rvalp = EBADF;
660         return (0);
661     }
662     vfsp = file->f_vnode->v_vfsp;
663     releasef((int)fd);

666     mutex_enter(&fsd_lock);

668     for (fsdi = list_head(&fsd_list); fsdi != NULL;
669         fsdi = list_next(&fsd_list, fsdi)) {
670         if (fsdi->fsdi_vfsp == vfsp)
671             break;
672     }
673     if (fsdi == NULL) {
674         *rvalp = ENOENT;
675         mutex_exit(&fsd_lock);
676         return (0);
677     }
678     rw_enter(&fsdi->fsdi_lock, RW_READER);
679     error = ddi_copyout(&fsdi->fsdi_param, &ioc->fsdioc_param,
680         sizeof (fsdi->fsdi_param), mode);
681     rw_exit(&fsdi->fsdi_lock);

683     mutex_exit(&fsd_lock);

686     if (error) {
687         return (EFAULT);
688     } else {
689         *rvalp = 0;
690         return (0);
691     }
692 }

694 static int
695 fsd_ioctl_get_info(fsd_ioc_t *ioc, int mode, int *rvalp)
696 {
697     fsd_info_t info;

699     mutex_enter(&fsd_lock);
700     info.fsdinf_enabled = fsd_enabled;
701     info.fsdinf_count = fsd_list_count;
702     info.fsdinf_omni_on = fsd_omni_param != NULL;
703     if (info.fsdinf_omni_on)
704         (void) memcpy(&info.fsdinf_omni_param, fsd_omni_param,
705             sizeof (info.fsdinf_omni_param));
706     mutex_exit(&fsd_lock);

708     if (ddi_copyout(&info, &ioc->fsdioc_info, sizeof (info), mode))
709         return (EFAULT);

711     *rvalp = 0;
712     return (0);
713 }

715 static int
716 fsd_ioctl_get_list(fsd_ioc_t *ioc, int mode, int *rvalp)
717 {
718     fsd_int_t *fsdi;
719     fsd_fs_t *fsdfs_list;
720     int i;
721     int ret = 0;

```

```

722     int64_t ioc_list_count;

724     *rvalp = 0;

726     /* Get data */
727     if (ddi_copyin(&ioc->fsdioc_list.count, &ioc_list_count,
728         sizeof (ioc_list_count), mode))
729         return (EFAULT);
730     if (ddi_copyin(&ioc->fsdioc_list.listp, &fsdfs_list,
731         sizeof (fsdfs_list), mode))
732         return (EFAULT);

735     mutex_enter(&fsd_lock);
736     if (ioc_list_count > fsd_list_count)
737         ioc_list_count = fsd_list_count;

739     /* Copyout */
740     if (ddi_copyout(&ioc_list_count, &ioc->fsdioc_list.count,
741         sizeof (ioc_list_count), mode)) {
742         ret = EFAULT;
743         goto out;
744     }
745     for (fsdi = list_head(&fsd_list), i = 0;
746         fsdi != NULL && i < ioc_list_count;
747         fsdi = list_next(&fsd_list, fsdi), i++) {
748         refstr_t *mntstr = vfs_getmntpoint(fsdi->fsdi_vfsp);
749         int len = strlen(refstr_value(mntstr));

751         rw_enter(&fsdi->fsdi_lock, RW_READER);
752         if (ddi_copyout(refstr_value(mntstr), fsdfs_list[i].fsdf_name,
753             len + 1, mode) ||
754             ddi_copyout(&fsdi->fsdi_param, &fsdfs_list[i].fsdf_param,
755                 sizeof (fsdi->fsdi_param), mode)) {
756             ret = EFAULT;
757         }
758         rw_exit(&fsdi->fsdi_lock);
759         refstr_rele(mntstr);

761         if (ret != 0)
762             break;
763     }

766 out:
767     mutex_exit(&fsd_lock);
768     return (ret);
769 }

771 static int
772 fsd_ioctl_disturb_off(fsd_ioc_t *ioc, int mode, int *rvalp)
773 {
774     file_t *file;
775     int64_t fd;

777     if (ddi_copyin(&ioc->fsdioc_mnt, &fd, sizeof (fd), mode))
778         return (EFAULT);

780     if ((file = getf((int)fd)) == NULL) {
781         *rvalp = EBADF;
782         return (0);
783     }

785     mutex_enter(&fsd_lock);
786     *rvalp = fsd_remove_disturber(file->f_vnode->v_vfsp);
787     releasef((int)fd);

```

```

788     mutex_exit(&fsd_lock);
790     return (0);
791 }

793 static int
794 fsd_ioctl_disturb_omni(fsd_ioc_t *ioc, int mode, int *rvalp)
795 {
796     fsd_t fsd;
797     int rv;

799     if (ddi_copyin(&ioc->fsdioc_param, &fsd, sizeof (fsd), mode))
800         return (EFAULT);

802     if ((rv = fsd_check_param(&fsd)) != 0) {
803         *rvalp = rv;
804         return (0);
805     }

807     mutex_enter(&fsd_lock);
808     if (fsd_omni_param == NULL)
809         fsd_omni_param = (fsd_t *)kmem_alloc(sizeof (*fsd_omni_param),
810             KM_SLEEP);
811     (void) memcpy(fsd_omni_param, &fsd, sizeof (*fsd_omni_param));
812     mutex_exit(&fsd_lock);

814     *rvalp = 0;
815     return (0);
816 }

819 static int
820 fsd_ioctl(dev_t dev, int cmd, intptr_t arg, int mode, cred_t *credp,
821     int *rvalp)
822 {
823     _NOTE(ARGUNUSED(dev));
824     _NOTE(ARGUNUSED(credp));

826     if (!fsd_enabled && cmd != FSD_ENABLE) {
827         *rvalp = ENOTACTIVE;
828         return (0);
829     }

831     switch (cmd) {
832     case FSD_ENABLE:
833         fsd_enable();
834         *rvalp = 0;
835         return (0);

837     case FSD_DISABLE:
838         fsd_disable();
839         *rvalp = 0;
840         return (0);

842     case FSD_GET_PARAM:
843         return (fsd_ioctl_get_param((fsd_ioc_t *)arg, mode, rvalp));

845     case FSD_DISTURB:
846         return (fsd_ioctl_disturb((fsd_ioc_t *)arg, mode, rvalp));

848     case FSD_DISTURB_OFF:
849         return (fsd_ioctl_disturb_off((fsd_ioc_t *)arg, mode, rvalp));

851     case FSD_DISTURB_OMNI:
852         return (fsd_ioctl_disturb_omni((fsd_ioc_t *)arg, mode, rvalp));

```

```

854     case FSD_DISTURB_OMNI_OFF:
855         mutex_enter(&fsd_lock);
856         if (fsd_omni_param != NULL)
857             kmem_free(fsd_omni_param, sizeof (*fsd_omni_param));
858         fsd_omni_param = NULL;
859         mutex_exit(&fsd_lock);

861         *rvalp = 0;
862         return (0);

864     case FSD_GET_LIST:
865         return (fsd_ioctl_get_list((fsd_ioc_t *)arg, mode, rvalp));

867     case FSD_GET_INFO:
868         return (fsd_ioctl_get_info((fsd_ioc_t *)arg, mode, rvalp));

870     default:
871         return (ENOTTY);
872     }
873 }

875 static struct cb_ops cb_ops = {
876     fsd_open,        /* open(9E) */
877     fsd_close,      /* close(9E) */
878     nodev,          /* strategy(9E) */
879     nodev,          /* print(9E) */
880     nodev,          /* dump(9E) */
881     nodev,          /* read(9E) */
882     nodev,          /* write(9E) */
883     fsd_ioctl,     /* ioctl(9E) */
884     nodev,          /* devmap(9E) */
885     nodev,          /* mmap(9E) */
886     nodev,          /* segmap(9E) */
887     nochpoll,      /* chpoll(9E) */
888     ddi_prop_op,   /* prop_op(9E) */
889     NULL,          /* streamtab(9E) */
890     D_MP | D_64BIT, /* cb_flag(9E) */
891     CB_REV,        /* cb_rev(9E) */
892     nodev,         /* aread(9E) */
893     nodev,         /* awrite(9E) */
894 };

896 static struct dev_ops dev_ops = {
897     DEVO_REV,      /* driver build version */
898     0,             /* reference count */
899     fsd_getinfo,   /* getinfo */
900     nulldev,
901     nulldev,      /* probe */
902     fsd_attach,   /* attach */
903     fsd_detach,   /* detach */
904     nodev,
905     &cb_ops,      /* cb_ops */
906     NULL,         /* bus_ops */
907     NULL,         /* power */
908     ddi_quiesce_not_needed, /* quiesce */
909 };

911 static struct modldrv modldrv = {
912     &mod_driverops, "Filesystem disturber", &dev_ops
913 };

915 static struct modlinkage modlinkage = {
916     MODREV_1, &modldrv, NULL
917 };

919 int

```

```
920 _init(void)
921 {
922     return (mod_install(&modlinkage));
923 }

925 int
926 _info(struct modinfo *modinfop)
927 {
928     return (mod_info(&modlinkage, modinfop));
929 }

931 int
932 _fini(void)
933 {
934     return (mod_remove(&modlinkage));
935 }
```

new/usr/src/uts/common/io/fsd/fsd.conf

1

\*\*\*\*\*

499 Tue Sep 3 18:09:41 2013

new/usr/src/uts/common/io/fsd/fsd.conf

fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 #
11 #
12 #
13 # Copyright 2013 Damian Bogel. All rights reserved.
14 #
15 name="fsd" parent="pseudo" instance=0;
```

```

*****
22750 Tue Sep 3 18:09:41 2013
new/usr/src/uts/common/sys/Makefile
fsh, fsd, libfsd, fsdadm
*****
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 (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 #
22 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
23 #

25 include $(SRC)/uts/Makefile.uts

27 FILEMODE=644

29 #
30 # Note that the following headers are present in the kernel but
31 # neither installed or shipped as part of the product:
32 # cpuid_drv.h: Private interface for cpuid consumers
33 # unix_bb_info.h: Private interface to kcov
34 #

36 i386_HDRS= \
37 agp/agpamd64gart_io.h \
38 agp/agpdefs.h \
39 agp/agpgart_impl.h \
40 agp/agpmaster_io.h \
41 agp/agptarget_io.h \
42 agpgart.h \
43 asy.h \
44 fd_debug.h \
45 fdc.h \
46 fdmedia.h \
47 mouse.h \
48 ucode.h

50 sparc_HDRS= \
51 mouse.h \
52 scsi/targets/ssddef.h \
53 $(MDESCHDRS)

55 # Generated headers
56 GENHDRS= \
57 priv_const.h \
58 priv_names.h \
59 usb/usbdevs.h

61 CHKHDRS= \

```

```

62 acpi_drv.h \
63 acct.h \
64 acctctl.h \
65 acl.h \
66 acl_impl.h \
67 aggr.h \
68 aggr_impl.h \
69 aio.h \
70 aio_impl.h \
71 aio_req.h \
72 aiocb.h \
73 ascii.h \
74 asynch.h \
75 atomic.h \
76 attr.h \
77 audio.h \
78 audioio.h \
79 autoconf.h \
80 auxv.h \
81 auxv_386.h \
82 auxv_SPARC.h \
83 avl.h \
84 avl_impl.h \
85 bitmap.h \
86 bitset.h \
87 bl.h \
88 blkdev.h \
89 bofi.h \
90 bofi_impl.h \
91 bpp_io.h \
92 bootstat.h \
93 brand.h \
94 buf.h \
95 bufmod.h \
96 bustypes.h \
97 byteorder.h \
98 callb.h \
99 callo.h \
100 cap_util.h \
101 cpucaps.h \
102 cpucaps_impl.h \
103 ccompile.h \
104 cdio.h \
105 cladm.h \
106 class.h \
107 clconf.h \
108 clock_impl.h \
109 cmlb.h \
110 cmn_err.h \
111 compress.h \
112 condvar.h \
113 condvar_impl.h \
114 conf.h \
115 consdev.h \
116 console.h \
117 consplat.h \
118 vt.h \
119 vtdaemon.h \
120 kd.h \
121 contract.h \
122 contract_impl.h \
123 copyops.h \
124 core.h \
125 corectl.h \
126 cpc_impl.h \
127 cpc_pcbe.h \

```

```

128     cpr.h                \|
129     cpupart.h           \|
130     cpuvar.h            \|
131     crc32.h             \|
132     cred.h              \|
133     cred_impl.h        \|
134     crtctl.h           \|
135     cryptmod.h         \|
136     csiiioctl.h       \|
137     ctf.h               \|
138     ctfs.h              \|
139     ctfs_impl.h       \|
140     ctf_api.h          \|
141     ctype.h            \|
142     cyclic.h           \|
143     cyclic_impl.h     \|
144     dacf.h             \|
145     dacf_impl.h       \|
146     damap.h            \|
147     damap_impl.h      \|
148     dc_ki.h            \|
149     ddi.h              \|
150     ddifm.h            \|
151     ddifm_impl.h      \|
152     ddi_hp.h           \|
153     ddi_hp_impl.h     \|
154     ddi_intr.h         \|
155     ddi_intr_impl.h   \|
156     ddi_impldefs.h    \|
157     ddi_implfuncs.h   \|
158     ddi_obsolete.h    \|
159     ddi_timer.h        \|
160     ddidevmap.h        \|
161     ddidmareq.h        \|
162     ddimapreq.h        \|
163     ddipropdefs.h     \|
164     dditypes.h         \|
165     debug.h            \|
166     des.h              \|
167     devctl.h           \|
168     devcache.h         \|
169     devcache_impl.h   \|
170     devfm.h            \|
171     devid_cache.h      \|
172     devinfo_impl.h    \|
173     devops.h           \|
174     devpolicy.h        \|
175     devpoll.h          \|
176     dirent.h           \|
177     disp.h             \|
178     dkbad.h            \|
179     dkio.h             \|
180     dklabel.h          \|
181     dl.h               \|
182     dlpi.h             \|
183     dld.h              \|
184     dld_impl.h         \|
185     dld_ioc.h         \|
186     dls.h              \|
187     dls_mgmt.h         \|
188     dls_impl.h         \|
189     dma_i8237A.h      \|
190     dnlc.h             \|
191     door.h             \|
192     door_data.h        \|
193     door_impl.h       \|

```

```

194     dtrace.h           \|
195     dtrace_impl.h     \|
196     dumpadm.h         \|
197     dumphdr.h         \|
198     ecppsys.h         \|
199     ecppio.h          \|
200     ecppreg.h         \|
201     ecppvar.h         \|
202     efi_partition.h   \|
203     elf.h              \|
204     elf_386.h         \|
205     elf_SPARC.h       \|
206     elf_notes.h       \|
207     elf_amd64.h       \|
208     elftypes.h        \|
209     emul64.h          \|
210     emul64cmd.h       \|
211     emul64var.h       \|
212     epm.h              \|
213     errno.h           \|
214     errorq.h          \|
215     errorq_impl.h     \|
216     esunddi.h         \|
217     ethernet.h        \|
218     euc.h              \|
219     eucliocntl.h      \|
220     exacctl.h         \|
221     exacctl_catalog.h \|
222     exacctl_impl.h    \|
223     exec.h            \|
224     exechnr.h         \|
225     extdirent.h       \|
226     fault.h           \|
227     fasttrap.h        \|
228     fasttrap_impl.h   \|
229     fbio.h            \|
230     fbuf.h            \|
231     fcntl.h           \|
232     fct.h              \|
233     fct_defines.h     \|
234     fctio.h           \|
235     fdbuffer.h        \|
236     fdio.h            \|
237     feature_tests.h   \|
238     fem.h              \|
239     file.h             \|
240     filio.h           \|
241     flock.h           \|
242     flock_impl.h      \|
243     fork.h            \|
244     fsd.h              \|
245     fsh.h              \|
246     fsh_impl.h        \|
247     fss.h              \|
248     fsspriocntl.h     \|
249     fsid.h            \|
250     fssnap.h          \|
251     fssnap_if.h       \|
252     fstyp.h           \|
253     ftrace.h          \|
254     fx.h               \|
255     fxpriocntl.h      \|
256     gfs.h              \|
257     gld.h              \|
258     gldpriv.h         \|
259     group.h           \|

```

```

260     hdio.h          \|
261     hook.h          \|
262     hook_event.h   \|
263     hook_impl.h    \|
264     hwconf.h       \|
265     ia.h            \|
266     iapriocntl.h   \|
267     ibpart.h       \|
268     id32.h         \|
269     idmap.h        \|
270     ieeefp.h       \|
271     id_space.h     \|
272     instance.h     \|
273     int_const.h    \|
274     int_fmtio.h    \|
275     int_limits.h   \|
276     int_types.h    \|
277     inttypes.h     \|
278     ioccom.h       \|
279     ioctl.h        \|
280     ipc.h          \|
281     ipc_impl.h     \|
282     ipc_rctl.h     \|
283     ipmi.h         \|
284     isa_defs.h     \|
285     iscsi_authclient.h \|
286     iscsi_authclientglue.h \|
287     iscsi_protocol.h \|
288     jiocntl.h      \|
289     kbd.h          \|
290     kbdreg.h       \|
291     kbio.h         \|
292     kcpic.h        \|
293     kdi.h          \|
294     kdi_impl.h     \|
295     kiconv.h       \|
296     kiconv_big5_utf8.h \|
297     kiconv_ck_common.h \|
298     kiconv_cp950hkscs_utf8.h \|
299     kiconv_emeal.h \|
300     kiconv_emea2.h \|
301     kiconv_euckr_utf8.h \|
302     kiconv_euctw_utf8.h \|
303     kiconv_gb18030_utf8.h \|
304     kiconv_gb2312_utf8.h \|
305     kiconv_hkscs_utf8.h \|
306     kiconv_ja.h    \|
307     kiconv_ja_jis_to_unicode.h \|
308     kiconv_ja_unicode_to_jis.h \|
309     kiconv_ko.h    \|
310     kiconv_latin1.h \|
311     kiconv_sc.h    \|
312     kiconv_tc.h    \|
313     kiconv_uhc_utf8.h \|
314     kiconv_utf8_big5.h \|
315     kiconv_utf8_cp950hkscs.h \|
316     kiconv_utf8_euckr.h \|
317     kiconv_utf8_euctw.h \|
318     kiconv_utf8_gb18030.h \|
319     kiconv_utf8_gb2312.h \|
320     kiconv_utf8_hkscs.h \|
321     kiconv_utf8_uhc.h \|
322     kidmap.h       \|
323     klpd.h         \|
324     klpw.h         \|
325     kmdb.h         \|

```

```

326     kmem.h         \|
327     kmem_impl.h   \|
328     kobj.h         \|
329     kobj_impl.h   \|
330     ksocket.h     \|
331     kstat.h       \|
332     kstr.h        \|
333     ksyms.h       \|
334     ksynch.h      \|
335     ldterm.h      \|
336     lgrp.h        \|
337     lgrp_user.h   \|
338     libc_kernel.h \|
339     link.h        \|
340     list.h        \|
341     list_impl.h   \|
342     llc1.h        \|
343     loadavg.h     \|
344     lock.h        \|
345     lockfs.h      \|
346     lockstat.h    \|
347     lofi.h        \|
348     log.h         \|
349     loginmux.h    \|
350     loginmux_impl.h \|
351     lwp.h         \|
352     lwp_timer_impl.h \|
353     lwp_upimutex_impl.h \|
354     lpif.h        \|
355     mac.h         \|
356     mac_client.h  \|
357     mac_client_impl.h \|
358     mac_ether.h   \|
359     mac_flow.h    \|
360     mac_flow_impl.h \|
361     mac_impl.h    \|
362     mac_provider.h \|
363     mac_soft_ring.h \|
364     mac_stat.h    \|
365     machelf.h     \|
366     map.h         \|
367     md4.h         \|
368     md5.h         \|
369     md5_consts.h  \|
370     mdi_impldefs.h \|
371     mem.h         \|
372     mem_config.h  \|
373     memlist.h     \|
374     mkdev.h       \|
375     mhd.h         \|
376     mi.h          \|
377     miiregs.h     \|
378     mixer.h       \|
379     mman.h        \|
380     mmapobj.h     \|
381     mntent.h      \|
382     mntio.h       \|
383     mnttab.h      \|
384     modctl.h      \|
385     mode.h        \|
386     model.h       \|
387     modhash.h     \|
388     modhash_impl.h \|
389     mount.h       \|
390     mouse.h       \|
391     msacct.h      \|

```

```

392     msg.h                \|
393     msg_impl.h          \|
394     msio.h              \|
395     msreg.h             \|
396     mtio.h              \|
397     multidata.h        \|
398     multidata_impl.h   \|
399     mutex.h             \|
400     nbmlock.h          \|
401     ndifm.h            \|
402     ndi_impldefs.h     \|
403     net80211.h         \|
404     net80211_crypto.h  \|
405     net80211_ht.h     \|
406     net80211_proto.h  \|
407     netconfig.h        \|
408     neti.h              \|
409     netstack.h         \|
410     nexusdefs.h       \|
411     note.h             \|
412     nvpair.h           \|
413     nvpair_impl.h     \|
414     objfs.h            \|
415     objfs_impl.h      \|
416     ontrap.h          \|
417     open.h             \|
418     openpromio.h      \|
419     panic.h           \|
420     param.h            \|
421     pathconf.h        \|
422     pathname.h        \|
423     pattn.h           \|
424     queue.h           \|
425     serializer.h      \|
426     pbio.h            \|
427     pccard.h          \|
428     pci.h              \|
429     pcie.h             \|
430     pci_impl.h        \|
431     pci_tools.h       \|
432     pcmcia.h          \|
433     pctype.h          \|
434     pfmod.h           \|
435     pg.h               \|
436     pghw.h            \|
437     physmem.h         \|
438     pkp_hash.h        \|
439     pm.h               \|
440     policy.h          \|
441     poll.h            \|
442     poll_impl.h       \|
443     pool.h            \|
444     pool_impl.h       \|
445     pool_pset.h       \|
446     port.h            \|
447     port_impl.h       \|
448     port_kernel.h     \|
449     portif.h          \|
450     ppmio.h           \|
451     pppt_ic_if.h      \|
452     pppt_ioctl.h      \|
453     priocntl.h        \|
454     priv.h            \|
455     priv_impl.h       \|
456     prnio.h           \|
457     proc.h            \|

```

```

458     processor.h       \|
459     procfs.h          \|
460     procset.h         \|
461     project.h         \|
462     protosw.h         \|
463     prsystem.h       \|
464     pset.h            \|
465     pshot.h          \|
466     ptem.h           \|
467     ptms.h           \|
468     ptyvar.h         \|
469     raidioctl.h     \|
470     ramdisk.h        \|
471     random.h         \|
472     rctl.h           \|
473     rctl_impl.h     \|
474     rds.h            \|
475     reboot.h        \|
476     refstr.h         \|
477     refstr_impl.h   \|
478     resource.h       \|
479     rliocntl.h      \|
480     rt.h             \|
481     rtpriocntl.h    \|
482     rwlock.h         \|
483     rwlock_impl.h   \|
484     rwstlock.h      \|
485     sad.h            \|
486     schedctl.h      \|
487     sdt.h            \|
488     select.h         \|
489     sem.h            \|
490     sem_impl.h       \|
491     sema_impl.h     \|
492     semaphore.h     \|
493     sendfile.h      \|
494     ser_sync.h       \|
495     session.h        \|
496     shal.h           \|
497     shal_consts.h   \|
498     sha2.h           \|
499     sha2_consts.h   \|
500     share.h          \|
501     shm.h            \|
502     shm_impl.h       \|
503     sid.h            \|
504     siginfo.h        \|
505     signal.h         \|
506     sleepq.h         \|
507     smbios.h         \|
508     smbios_impl.h   \|
509     subject.h        \|
510     socket.h         \|
511     socket_impl.h   \|
512     socket_proto.h  \|
513     socketvar.h     \|
514     sockfilter.h    \|
515     sockio.h         \|
516     soundcard.h     \|
517     squeue.h         \|
518     squeue_impl.h   \|
519     srn.h            \|
520     sservice.h       \|
521     stat.h           \|
522     statfs.h         \|
523     statvfs.h       \|

```



```

524     stdbool.h          \|
525     stdint.h           \|
526     stermio.h          \|
527     stmf.h             \|
528     stmf_defines.h    \|
529     stmf_ioctl.h      \|
530     stmf_sbd_ioctl.h  \|
531     stream.h           \|
532     strft.h            \|
533     strlog.h           \|
534     strmddep.h         \|
535     stropts.h          \|
536     strredir.h         \|
537     strstat.h          \|
538     strsubr.h          \|
539     strsun.h           \|
540     strtty.h           \|
541     sunddi.h           \|
542     sunldi.h           \|
543     sunldi_impl.h     \|
544     sunmdi.h           \|
545     sunndi.h           \|
546     sunos_dhcp_class.h \|
547     sunpm.h            \|
548     suntpl.h           \|
549     suntty.h           \|
550     swap.h             \|
551     synch.h            \|
552     sysdc.h            \|
553     sysdc_impl.h      \|
554     syscall.h          \|
555     sysconf.h          \|
556     sysconfig.h       \|
557     sysevent.h         \|
558     sysevent_impl.h  \|
559     sysinfo.h          \|
560     syslog.h           \|
561     sysmacros.h        \|
562     sysmsg_impl.h     \|
563     systeminfo.h      \|
564     system.h           \|
565     task.h             \|
566     taskq.h            \|
567     taskq_impl.h      \|
568     t_kuser.h          \|
569     t_lock.h           \|
570     telioctl.h        \|
571     termio.h           \|
572     termios.h          \|
573     termiox.h          \|
574     thread.h           \|
575     ticlts.h           \|
576     ticots.h           \|
577     ticotsord.h       \|
578     tihdr.h            \|
579     time.h              \|
580     time_impl.h        \|
581     time_std_impl.h   \|
582     timeb.h            \|
583     timer.h            \|
584     times.h            \|
585     timex.h            \|
586     timod.h            \|
587     tirdwr.h           \|
588     tiuser.h           \|
589     tl.h                \|

```

```

590     tnf.h               \|
591     tnf_com.h           \|
592     tnf_probe.h        \|
593     tnf_writer.h       \|
594     todio.h            \|
595     tpicommon.h        \|
596     ts.h                \|
597     tspriocntl.h       \|
598     ttcompat.h         \|
599     ttold.h             \|
600     tty.h               \|
601     ttychars.h         \|
602     ttydev.h           \|
603     tuneable.h         \|
604     turnstile.h        \|
605     types.h             \|
606     types32.h          \|
607     tzfile.h           \|
608     u8_textprep.h      \|
609     u8_textprep_data.h \|
610     uadmin.h           \|
611     ucred.h            \|
612     uio.h               \|
613     ulimit.h           \|
614     un.h                \|
615     unistd.h           \|
616     user.h             \|
617     ustat.h            \|
618     utime.h            \|
619     utsname.h          \|
620     utssys.h           \|
621     uuid.h              \|
622     va_impl.h          \|
623     va_list.h          \|
624     var.h               \|
625     varargs.h          \|
626     vfs.h               \|
627     vfs_opreg.h        \|
628     vfstab.h           \|
629     vgareg.h           \|
630     videodev2.h        \|
631     visual_io.h        \|
632     vlan.h             \|
633     vm.h                \|
634     vm_usage.h         \|
635     vmem.h             \|
636     vmem_impl.h        \|
637     vmsystem.h         \|
638     vnic.h              \|
639     vnic_impl.h        \|
640     vnode.h            \|
641     vscan.h            \|
642     vtoc.h             \|
643     vtrace.h           \|
644     vuid_event.h       \|
645     vuid_wheel.h       \|
646     vuid_queue.h       \|
647     vuid_state.h       \|
648     vuid_store.h       \|
649     wait.h             \|
650     waitq.h            \|
651     wanboot_impl.h     \|
652     watchpoint.h       \|
653     winlockio.h        \|
654     zcons.h            \|
655     zone.h              \|

```

```

656      xti_inet.h      \
657      xti_osi.h      \
658      xti_xtiopt.h   \
659      zmod.h         \

661 HDRS=              \
662      $(GENHDRS)     \
663      $(CHKHDRS)     \

665 AUDIOHDRS=        \
666      ac97.h         \
667      audio_common.h \
668      audio_driver.h \
669      audio_oss.h    \
670      g711.h         \

672 AVHDRS=            \
673      iec61883.h    \

675 BSCHDRS=          \
676      bscbus.h      \
677      bscv_impl.h   \
678      lom_ebuscodes.h \
679      lom_io.h       \
680      lom_priv.h    \
681      lombus.h      \

683 MDESCHDRS=        \
684      mdesc.h        \
685      mdesc_impl.h  \

687 CPUDRVHDRS=       \
688      cpudrv.h      \

690 CRYPTOHDRS=       \
691      elfsign.h     \
692      ioctl.h       \
693      ioctladmin.h  \
694      common.h      \
695      impl.h        \
696      spi.h         \
697      api.h         \
698      ops_impl.h    \
699      sched_impl.h  \

701 DCAMHDRS=         \
702      dcam1394_io.h \

704 IBHDRS=           \
705      ib_types.h    \
706      ib_pkt_hdrs.h \

708 IBTLHDRS=         \
709      ibtl_types.h  \
710      ibtl_status.h \
711      ibti.h        \
712      ibti_cm.h     \
713      ibci.h        \
714      ibti_common.h \
715      ibvti.h       \
716      ibtl_ci_types.h \

718 IBTLIMPLHDRS=     \
719      ibtl_util.h   \

721 IBNEXHDRS=        \

```

```

722      ibnex_devctl.h \

724 IBMFHDRS=         \
725      ibmf.h        \
726      ibmf_msg.h    \
727      ibmf_saa.h    \
728      ibmf_utils.h  \

730 IBMGTHDRS=        \
731      ib_dm_attr.h  \
732      ib_mad.h      \
733      sm_attr.h     \
734      sa_recs.h     \

736 IBDHDRS=          \
737      ibd.h         \

739 OFHDRS=           \
740      ofa_solaris.h \
741      ofed_kernel.h \

743 RDMAHDRS=         \
744      ib_addr.h     \
745      ib_user_mad.h \
746      ib_user_sa.h  \
747      ib_user_verbs.h \
748      ib_verbs.h    \
749      rdma_cm.h     \
750      rdma_user_cm.h \

752 SOL_UVERBSHDRS=   \
753      sol_uverbs.h  \
754      sol_uverbs2ucma.h \
755      sol_uverbs_comp.h \
756      sol_uverbs_hca.h \
757      sol_uverbs_qp.h \
758      sol_uverbs_event.h \

760 SOL_UMADHDRS=     \
761      sol_umad.h    \

763 SOL_UCMAHDRS=     \
764      sol_ucma.h    \
765      sol_rdma_user_cm.h \

767 SOL_OFSHDRS=      \
768      sol_cma.h     \
769      sol_ib_cma.h  \
770      sol_ofs_common.h \
771      sol_kverb_impl.h \

773 TAVORHDRS=        \
774      tavor_ioctl.h \

776 HERMONHDRS=       \
777      hermon_ioctl.h \

779 MLNXHDRS=         \
780      mlnx_umap.h  \

782 IDMHDRS=          \
783      idm.h         \
784      idm_impl.h    \
785      idm_so.h      \
786      idm_text.h    \
787      idm_transport.h \

```

```

788     idm_conn_sm.h
790 ISCSITHDRS= \
791     radius_packet.h \
792     radius_protocol.h \
793     chap.h \
794     isns_protocol.h \
795     iscsi_if.h \
796     iscsit_common.h
798 ISOHDRS= \
799     signal_iso.h
801 DERIVED_LVMHDRS= \
802     md_mdiox.h \
803     md_basic.h \
804     mdmed.h \
805     md_mhdx.h \
806     mdmn_commd.h
808 LVMHDRS= \
809     md_convert.h \
810     md_crc.h \
811     md_hotspares.h \
812     md_mddb.h \
813     md_mirror.h \
814     md_mirror_shared.h \
815     md_names.h \
816     md_notify.h \
817     md_raid.h \
818     md_rename.h \
819     md_sp.h \
820     md_stripe.h \
821     md_trans.h \
822     mdio.h \
823     mdvar.h
825 ALL_LVMHDRS= \
826     $(LVMHDRS) \
827     $(DERIVED_LVMHDRS)
829 FMHDRS= \
830     protocol.h \
831     util.h
833 FMFSHDRS= \
834     zfs.h
836 FMIOHDRS= \
837     ddi.h \
838     disk.h \
839     pci.h \
840     scsi.h \
841     sun4upci.h \
842     opl_mc_fm.h
844 FSHDRS= \
845     autofs.h \
846     cacheofs_dir.h \
847     cacheofs_dlog.h \
848     cacheofs_filegrp.h \
849     cacheofs_fs.h \
850     cacheofs_fscache.h \
851     cacheofs_ioctl.h \
852     cacheofs_log.h \
853     decomp.h

```

```

854     dv_node.h \
855     sdev_impl.h \
856     fifonode.h \
857     hsfs_isospec.h \
858     hsfs_node.h \
859     hsfs_rrip.h \
860     hsfs_spec.h \
861     hsfs_susp.h \
862     lofs_info.h \
863     lofs_node.h \
864     mntdata.h \
865     namenode.h \
866     pc_dir.h \
867     pc_fs.h \
868     pc_label.h \
869     pc_node.h \
870     pxfk_ki.h \
871     snode.h \
872     swapnode.h \
873     tmp.h \
874     tmpnode.h \
875     udf_inode.h \
876     udf_volume.h \
877     ufs_acl.h \
878     ufs_bio.h \
879     ufs_filio.h \
880     ufs_fs.h \
881     ufs_fsdire.h \
882     ufs_inode.h \
883     ufs_lockfs.h \
884     ufs_log.h \
885     ufs_mount.h \
886     ufs_panic.h \
887     ufs_prot.h \
888     ufs_quota.h \
889     ufs_snap.h \
890     ufs_trans.h \
891     zfs.h \
892     zut.h
894 PCMCIAHDRS= \
895     pcata.h \
896     pcser_conf.h \
897     pcser_io.h \
898     pcser_reg.h \
899     pcser_manuspec.h \
900     pcser_var.h
902 SCSIHDRS= \
903     scsi.h \
904     scsi_address.h \
905     scsi_ctl.h \
906     scsi_fm.h \
907     scsi_params.h \
908     scsi_pkt.h \
909     scsi_resource.h \
910     scsi_types.h \
911     scsi_watch.h
913 SCSSICONFHDRS= \
914     autoconf.h \
915     device.h
917 SCSSIGENHDRS= \
918     commands.h \
919     dad_mode.h

```

```

920      inquiry.h      \
921      message.h     \
922      mode.h        \
923      persist.h     \
924      sense.h       \
925      sff_frames.h  \
926      smp_frames.h  \
927      status.h      \

929 SCSIIMPLHDRS=    \
930      commands.h   \
931      inquiry.h    \
932      mode.h       \
933      scsi_reset_notify.h \
934      scsi_sas.h   \
935      sense.h      \
936      services.h  \
937      smp_transport.h \
938      spc3_types.h \
939      status.h     \
940      transport.h  \
941      types.h      \
942      uscsi.h      \
943      usmp.h       \

945 SCSTITARGETSHDRS= \
946      ses.h        \
947      sesio.h     \
948      sgendef.h   \
949      stdef.h     \
950      sddef.h     \
951      smp.h       \

953 SCSIADHDRS=

955 SCASICADHDRS=

957 SCSIISCSIHDRS=  \
958      iscsi_door.h \
959      iscsi_if.h   \

961 SCSEIVHCIHDRS=  \
962      scsi_vhci.h  \
963      mpapi_impl.h \
964      mpapi_scsi_vhci.h \

966 SDCARDHDRS=     \
967      sda.h        \
968      sda_impl.h   \
969      sda_ioctl.h  \

971 FC4HDRS=        \
972      fc_transport.h \
973      linkapp.h     \
974      fc.h          \
975      fcp.h         \
976      fcal_transport.h \
977      fcal.h        \
978      fcal_linkapp.h \
979      fcio.h        \

981 FCHDRS=         \
982      fc.h         \
983      fcio.h       \
984      fc_types.h  \
985      fc_appif.h

```

```

987 FCIMPLHDRS=     \
988      fc_error.h  \
989      fcph.h      \

991 FCULPHDRS=      \
992      fcp_util.h  \
993      fcsml.h     \

995 SATAGENHDRS=    \
996      sata_hba.h  \
997      sata_defs.h \
998      sata_cfgadm.h \

1000 SYSEVENTHDRS=   \
1001      ap_driver.h \
1002      dev.h       \
1003      domain.h   \
1004      dr.h        \
1005      env.h       \
1006      eventdefs.h \
1007      imp.h       \
1008      pwrctl.h   \
1009      svm.h       \
1010      vrrp.h     \

1012 CONTRACTHDRS=   \
1013      process.h   \
1014      process_impl.h \
1015      device.h    \
1016      device_impl.h \

1018 USBHDRS=        \
1019      usba.h      \
1020      usbai.h     \

1022 UWBHDRS=        \
1023      uwb.h       \
1024      uwbai.h     \

1026 UWBAHDRS=      \
1027      uwba.h     \

1029 USBAUDHDRS=    \
1030      usb_audio.h \

1032 USBHUBDHDRS=   \
1033      hub.h       \
1034      hubd_impl.h \

1036 USBHIDHDRS=    \
1037      hid.h       \

1039 USBHWARDHDRS=  \
1040      hwarc.h     \

1042 USBMSHDRS=     \
1043      usb_bulkonly.h \
1044      usb_cbi.h   \

1046 USBPRNHDRS=   \
1047      usb_printer.h \

1049 USBDCDCHDRS=  \
1050      usb_cdc.h

```

```

1052 USBVIDHDRS= \
1053     usbvc.h

1055 USBWCMHDRS= \
1056     usbwcm.h

1058 UGENHDRS= \
1059     usb_ugen.h

1061 HOTPLUGHDRS= \
1062     hpcsvc.h \
1063     hpctrl.h

1065 HOTPLUGPCIHDRS= \
1066     pcicfg.h \
1067     pcihp.h

1069 RSMHDRS= \
1070     rsm.h \
1071     rsm_common.h \
1072     rsmapi_common.h \
1073     rsmapi.h \
1074     rsmapi_driver.h \
1075     rsmka_path_int.h

1077 TSOLHDRS= \
1078     label.h \
1079     label_macro.h \
1080     priv.h \
1081     tndb.h \
1082     tsyscall.h

1084 I1394HDRS= \
1085     cmd1394.h \
1086     id1394.h \
1087     ieee1212.h \
1088     ieee1394.h \
1089     ixl1394.h \
1090     sl394_impl.h \
1091     tl394.h

1093 # "cmdk" headers used on sparc
1094 SDKTPHDRS= \
1095     dadkio.h \
1096     fdisk.h

1098 # "cmdk" headers used on i386
1099 DKTPHDRS= \
1100     altsctr.h \
1101     bbh.h \
1102     cm.h \
1103     cmdev.h \
1104     cmdk.h \
1105     cmpkt.h \
1106     controller.h \
1107     dadev.h \
1108     dadk.h \
1109     dadkio.h \
1110     fctypes.h \
1111     fdisk.h \
1112     flowctrl.h \
1113     gda.h \
1114     quetypes.h \
1115     queue.h \
1116     tgcom.h \
1117     tgdk.h

```

```

1119 # "pc" header files used on i386
1120 PCHDRS= \
1121     avintr.h \
1122     dma_engine.h \
1123     i8272A.h \
1124     pcic_reg.h \
1125     pcic_var.h \
1126     pic.h \
1127     pit.h \
1128     rtc.h

1130 NXGEHDRS= \
1131     nxge.h \
1132     nxge_common.h \
1133     nxge_common_impl.h \
1134     nxge_defs.h \
1135     nxge_hw.h \
1136     nxge_impl.h \
1137     nxge_ipp.h \
1138     nxge_ipp_hw.h \
1139     nxge_mac.h \
1140     nxge_mac_hw.h \
1141     nxge_fflp.h \
1142     nxge_fflp_hw.h \
1143     nxge_mii.h \
1144     nxge_rxdma.h \
1145     nxge_rxdma_hw.h \
1146     nxge_txc.h \
1147     nxge_txc_hw.h \
1148     nxge_txdma.h \
1149     nxge_txdma_hw.h \
1150     nxge_virtual.h \
1151     nxge_espc.h

1153 include Makefile.syshdrs

1155 dcam/%.check: dcam/%.h
1156     $(DOT_H_CHECK)

1158 CHECKHDRS= \
1159     $(MACH)_HDRS:%.h=%.check \
1160     $(AUDIOHDRS:%.h=audio/%.check) \
1161     $(AVHDRS:%.h=av/%.check) \
1162     $(BSCHDRS:%.h=%.check) \
1163     $(CHKHDRS:%.h=%.check) \
1164     $(CPUDRVHDRS:%.h=%.check) \
1165     $(CRYPTOHDRS:%.h=crypto/%.check) \
1166     $(DCAMHDRS:%.h=dcam/%.check) \
1167     $(FC4HDRS:%.h=fc4/%.check) \
1168     $(FCHDRS:%.h=fibre-channel/%.check) \
1169     $(FCIMPLHDRS:%.h=fibre-channel/impl/%.check) \
1170     $(FCULPHDRS:%.h=fibre-channel/ulp/%.check) \
1171     $(IBHDRS:%.h=ib/%.check) \
1172     $(IBDHDRS:%.h=ib/clients/ibd/%.check) \
1173     $(IBTLHDRS:%.h=ib/ibt1/%.check) \
1174     $(IBTLIMPLHDRS:%.h=ib/ibt1/impl/%.check) \
1175     $(IBNEXHDRS:%.h=ib/ibnex/%.check) \
1176     $(IBMGTHDRS:%.h=ib/mgt/%.check) \
1177     $(IBMFHDRS:%.h=ib/mgt/ibmf/%.check) \
1178     $(OFHDRS:%.h=ib/clients/of/%.check) \
1179     $(RDMAHDRS:%.h=ib/clients/of/rdma/%.check) \
1180     $(SOL_UVERBSHDRS:%.h=ib/clients/of/sol_uverbs/%.check) \
1181     $(SOL_UCMAHDRS:%.h=ib/clients/of/sol_ucma/%.check) \
1182     $(SOL_OFSHDRS:%.h=ib/clients/of/sol_ofs/%.check) \
1183     $(TAVORHDRS:%.h=ib/adapters/tavor/%.check) \

```

```

1184 $(HERMONHDRS:%.h=ib/adapters/hermon/%.check) \
1185 $(MLNXHDRS:%.h=ib/adapters/%.check) \
1186 $(IDMHDRS:%.h=idm/%.check) \
1187 $(ISCSIHDRS:%.h=iscsi/%.check) \
1188 $(ISCSITHDRS:%.h=iscsit/%.check) \
1189 $(ISOHDRS:%.h=iso/%.check) \
1190 $(FMHDRS:%.h=fm/%.check) \
1191 $(FMFSDHDRS:%.h=fm/fs/%.check) \
1192 $(FMIOHDRS:%.h=fm/io/%.check) \
1193 $(FSHDRS:%.h=fs/%.check) \
1194 $(LVMHDRS:%.h=lvm/%.check) \
1195 $(PCMCIAHDRS:%.h=pcmcia/%.check) \
1196 $(SCSIHDRS:%.h=scsi/%.check) \
1197 $(SCSIADHDRS:%.h=scsi/adapters/%.check) \
1198 $(SCSICONFHDRS:%.h=scsi/conf/%.check) \
1199 $(SCSIIMPLHDRS:%.h=scsi/impl/%.check) \
1200 $(SCSIISCSIHDRS:%.h=scsi/adapters/%.check) \
1201 $(SCSIGHDRS:%.h=scsi/generic/%.check) \
1202 $(SCSITARGETSHDRS:%.h=scsi/targets/%.check) \
1203 $(SCSIVHCIHDRS:%.h=scsi/adapters/%.check) \
1204 $(SATAGENHDRS:%.h=sata/%.check) \
1205 $(SDCARDHDRS:%.h=sdcard/%.check) \
1206 $(SYSEVENTHDRS:%.h=sysevent/%.check) \
1207 $(CONTRACTHDRS:%.h=contract/%.check) \
1208 $(USBAUDHDRS:%.h=usb/clients/audio/%.check) \
1209 $(USBHUBDHDRS:%.h=usb/hubd/%.check) \
1210 $(USBHIDHDRS:%.h=usb/clients/hid/%.check) \
1211 $(USBHWARCHDRS:%.h=usb/clients/hwarc/%.check) \
1212 $(USBMSHDRS:%.h=usb/clients/mass_storage/%.check) \
1213 $(USBPRNHDRS:%.h=usb/clients/printer/%.check) \
1214 $(USBDCDCHDRS:%.h=usb/clients/usbcdc/%.check) \
1215 $(USBVIDHDRS:%.h=usb/clients/video/usbvc/%.check) \
1216 $(USBWCMHDRS:%.h=usb/clients/usbinput/usbwcm/%.check) \
1217 $(UGENHDRS:%.h=usb/clients/ugen/%.check) \
1218 $(USBHDRS:%.h=usb/%.check) \
1219 $(UWBHDRS:%.h=uwb/%.check) \
1220 $(UWBAHDRS:%.h=uwb/uwba/%.check) \
1221 $(I1394HDRS:%.h=1394/%.check) \
1222 $(RSMHDRS:%.h=rsm/%.check) \
1223 $(TSOLHDRS:%.h=tso1/%.check) \
1224 $(NXGEHDRS:%.h=nxge/%.check)

```

```
1227 .KEEP_STATE:
```

```

1229 .PARALLEL: \
1230 $(CHECKHDRS) \
1231 $(ROOTHDRS) \
1232 $(ROOTAUDHDRS) \
1233 $(ROOTAVHDRS) \
1234 $(ROOTCRYPTOHDERS) \
1235 $(ROOTDCAMHDRS) \
1236 $(ROOTISOHDRS) \
1237 $(ROOTIDMHDRS) \
1238 $(ROOTISCSIHDRS) \
1239 $(ROOTISCSITHDRS) \
1240 $(ROOTFC4HDRS) \
1241 $(ROOTFCHDRS) \
1242 $(ROOTFCIMPLHDRS) \
1243 $(ROOTFCULPHDRS) \
1244 $(ROOTFMHDRS) \
1245 $(ROOTFMIOHDRS) \
1246 $(ROOTFMFSDHDRS) \
1247 $(ROOTFSDHDRS) \
1248 $(ROOTIBDHDRS) \
1249 $(ROOTIBHDRS)

```

```

1250 $(ROOTIBTLHDRS) \
1251 $(ROOTIBTLIMPLHDRS) \
1252 $(ROOTIBNEXHDRS) \
1253 $(ROOTIBMGTHDRS) \
1254 $(ROOTIBMFDHDRS) \
1255 $(ROOTOFHDRS) \
1256 $(ROOTRDMADHDRS) \
1257 $(ROOTSOL_OFSDHDRS) \
1258 $(ROOTSOL_UMADHDRS) \
1259 $(ROOTSOL_UVERBSHDRS) \
1260 $(ROOTSOL_UCMAHDRS) \
1261 $(ROOTTAVORHDRS) \
1262 $(ROOTTHERMONHDRS) \
1263 $(ROOTMLNXHDRS) \
1264 $(ROOTLVMHDRS) \
1265 $(ROOTPCMCIAHDRS) \
1266 $(ROOTSCSIHDRS) \
1267 $(ROOTSCSIADHDRS) \
1268 $(ROOTSCSICONFHDRS) \
1269 $(ROOTSCSIIISCSIHDRS) \
1270 $(ROOTSCSIGHDRS) \
1271 $(ROOTSCSIIMPLHDRS) \
1272 $(ROOTSCSIVHCIHDRS) \
1273 $(ROOTSDCARDHDRS) \
1274 $(ROOTSYSEVENTHDRS) \
1275 $(ROOTCONTRACTHDRS) \
1276 $(ROOTUSBHDRS) \
1277 $(ROOTUWBHDRS) \
1278 $(ROOTUWBAHDRS) \
1279 $(ROOTUSBAUDHDRS) \
1280 $(ROOTUSBHUBDHDRS) \
1281 $(ROOTUSBHIDHDRS) \
1282 $(ROOTUSBHRCHDRS) \
1283 $(ROOTUSBMSHDRS) \
1284 $(ROOTUSBPRNHDRS) \
1285 $(ROOTUSBDCDCHDRS) \
1286 $(ROOTUSBVIDHDRS) \
1287 $(ROOTUSBWCMHDRS) \
1288 $(ROOTUGENHDRS) \
1289 $(ROOT1394HDRS) \
1290 $(ROOTHOTPLUGHDRS) \
1291 $(ROOTHOTPLUGPCIHDRS) \
1292 $(ROOTRSMHDRS) \
1293 $(ROOTTSOLHDRS) \
1294 $( $(MACH)_ROOTHDRS)

```

```

1297 install_h: \
1298 $(ROOTDIRS) \
1299 LVMDERIVED_H \
1300 .WAIT \
1301 $(ROOTHDRS) \
1302 $(ROOTAUDHDRS) \
1303 $(ROOTAVHDRS) \
1304 $(ROOTCRYPTOHDERS) \
1305 $(ROOTDCAMHDRS) \
1306 $(ROOTISOHDRS) \
1307 $(ROOTIDMHDRS) \
1308 $(ROOTISCSIHDRS) \
1309 $(ROOTISCSITHDRS) \
1310 $(ROOTFC4HDRS) \
1311 $(ROOTFCHDRS) \
1312 $(ROOTFCIMPLHDRS) \
1313 $(ROOTFCULPHDRS) \
1314 $(ROOTFMHDRS) \
1315 $(ROOTFMFSDHDRS)

```

```

1316 $(ROOTFMIOHDRS) \
1317 $(ROOTFSHDRS) \
1318 $(ROOTIBDHDRS) \
1319 $(ROOTIBHDRS) \
1320 $(ROOTIBTLHDRS) \
1321 $(ROOTIBTLIMPLHDRS) \
1322 $(ROOTIBNEXHDRS) \
1323 $(ROOTIBMGTHDRS) \
1324 $(ROOTIBMFHDRS) \
1325 $(ROOTOFHDRS) \
1326 $(ROOTRDMHDRS) \
1327 $(ROOTSOL_OFSHDRS) \
1328 $(ROOTSOL_UMADHDRS) \
1329 $(ROOTSOL_UVERBSHDRS) \
1330 $(ROOTSOL_UCMAHDRS) \
1331 $(ROOTTAVORHDRS) \
1332 $(ROOTTHERMONHDRS) \
1333 $(ROOTMLNXHDRS) \
1334 $(ROOTLVMHDRS) \
1335 $(ROOTPCMCIAHDRS) \
1336 $(ROOTSCSIHDRS) \
1337 $(ROOTSCSIADHDRS) \
1338 $(ROOTSCSIIISCSIHDRS) \
1339 $(ROOTSCSICONFHDRS) \
1340 $(ROOTSCSIGENHDRS) \
1341 $(ROOTSCSIIMPLHDRS) \
1342 $(ROOTSCSIVHCIHDRS) \
1343 $(ROOTSDCARDHDRS) \
1344 $(ROOTSYSEVENTHDRS) \
1345 $(ROOTCONTRACTHDRS) \
1346 $(ROOTUWBHDRS) \
1347 $(ROOTUWBAHDRS) \
1348 $(ROOTUSBHDRS) \
1349 $(ROOTUSBAUDHDRS) \
1350 $(ROOTUSBHUBDHDRS) \
1351 $(ROOTUSBHIDHDRS) \
1352 $(ROOTUSBHRC HDRS) \
1353 $(ROOTUSBMSHDRS) \
1354 $(ROOTUSBPRNHDRS) \
1355 $(ROOTUSBDCDCHDRS) \
1356 $(ROOTUSBVIDHDRS) \
1357 $(ROOTUSBWCMHDRS) \
1358 $(ROOTUGENHDRS) \
1359 $(ROOT1394HDRS) \
1360 $(ROOTHOTPLUGHDRS) \
1361 $(ROOTHOTPLUGPCIHDRS) \
1362 $(ROOTRSMHDRS) \
1363 $(ROOTTSOLHDRS) \
1364 $( $(MACH)_ROOTHDRS)

1366 all_h: $(GENHDRS)

1368 priv_const.h: $(PRIVS_AWK) $(PRIVS_DEF)
1369 $(NAWK) -f $(PRIVS_AWK) < $(PRIVS_DEF) -v privhfile=$@

1371 priv_names.h: $(PRIVS_AWK) $(PRIVS_DEF)
1372 $(NAWK) -f $(PRIVS_AWK) < $(PRIVS_DEF) -v pubhfile=$@

1374 usb/usbdevs.h: $(USBDEVS_AWK) $(USBDEVS_DATA)
1375 $(NAWK) -f $(USBDEVS_AWK) $(USBDEVS_DATA) -H > $@

1377 LVMDERIVED_H:
1378 cd $(SRC)/uts/common/sys/lvm; pwd; $(MAKE)

1380 clean:
1381 $(RM) $(GENHDRS)

```

```

1383 clobber: clean

1385 check: $(CHECKHDRS)

1387 FRC:

1389 # EXPORT DELETE START
1390 EXPORT_SRC:
1391 $(RM) wanboot_impl.h+ Makefile+
1392 sed -e "/EXPORT DELETE START/,/EXPORT DELETE END/d" \
1393 < wanboot_impl.h > wanboot_impl.h+
1394 $(MV) wanboot_impl.h+ wanboot_impl.h
1395 sed -e "/^# EXPORT DELETE START/,/^# EXPORT DELETE END/d" \
1396 < Makefile > Makefile+
1397 $(RM) Makefile
1398 $(MV) Makefile+ Makefile
1399 $(CHMOD) 444 Makefile wanboot_impl.h
1400 # EXPORT DELETE END

```

```

*****
2223 Tue Sep 3 18:09:42 2013
new/usr/src/uts/common/sys/fsd.h
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel. All rights reserved.
14 */

16 #ifndef _FSD_H
17 #define _FSD_H

19 /*
20  * filesystem disturber header file
21 */

23 #ifdef __cplusplus
24 extern "C" {
25 #endif

27 #include <sys/errno.h>
28 #include <sys/ksynch.h>
29 #include <sys/list.h>
30 #include <sys/param.h>
31 #include <sys/types.h>
32 #include <sys/vfs.h>

35 #define FSD_DEV_PATH    "/dev/fsd"

37 /*
38  * Commands for ioctl().
39 */
40 #define FSDIOC (('f' << 24) | ('s' << 16) | ('d' << 8))
41 #define FSD_GET_PARAM      (FSDIOC | 1)
42 #define FSD_ENABLE        (FSDIOC | 2)
43 #define FSD_DISABLE       (FSDIOC | 3)
44 #define FSD_DISTURB       (FSDIOC | 4)
45 #define FSD_DISTURB_OFF   (FSDIOC | 5)
46 #define FSD_DISTURB_OMNI  (FSDIOC | 6)
47 #define FSD_DISTURB_OMNI_OFF (FSDIOC | 7)
48 #define FSD_GET_LIST      (FSDIOC | 8)
49 #define FSD_GET_INFO      (FSDIOC | 9)

52 /*
53  * Parameters description:
54  * "read less"
55  *   Makes a VOP_READ() call read n (from a given range) bytes less than it
56  *   was requested.
57 */
58 typedef struct fsd {
59     uint64_t    read_less_chance;
60     uint64_t    read_less_r[2]; /* range */
61 } fsd_t;

```

```

63 typedef struct fsd_info {
64     uint64_t    fsdinf_enabled; /* fsd enabled */
65     uint64_t    fsdinf_count; /* disturbers installed */
66     uint64_t    fsdinf_omni_on; /* omnipresent disturber on */
67     fsd_t        fsdinf_omni_param; /* omnipresent dist. params */
68 } fsd_info_t;

70 typedef struct fsd_dis {
71     int64_t    fsdd_mnt;
72     fsd_t        fsdd_param;
73 } fsd_dis_t;

75 typedef struct fsd_fs {
76     fsd_t        fsdf_param;
77     uint8_t    fsdf_name[MAXPATHLEN];
78 } fsd_fs_t;

80 typedef union fsd_ioc { /* Used with: */
81     fsd_info_t    fsdioc_info; /* _GET_INFO */
82     fsd_dis_t    fsdioc_dis; /* _DISTURB */
83     fsd_t        fsdioc_param; /* _GET_PARAM out, _DISTURB_OMNI */
84     int64_t    fsdioc_mnt; /* _DISTURB_OFF, _GET_PARAM in */
85     struct {
86         int64_t    count;
87         uint64_t    listp;
88     } fsdioc_list; /* _GET_LIST */
89 } fsd_ioc_t;

91 #ifdef __cplusplus
92 }
93 #endif

95 #endif /* _FSD_H */

```



```

*****
2076 Tue Sep 3 18:09:42 2013
new/usr/src/uts/common/sys/fsh.h
fsh, fsd, libfsd, fsdadm
*****
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 Damian Bogel. All rights reserved.
14 */

16 #ifndef _FSH_H
17 #define _FSH_H

19 #include <sys/id_space.h>
20 #include <sys/types.h>
21 #include <sys/vfs.h>
22 #include <sys/vnode.h>

24 #ifdef __cplusplus
25 extern "C" {
26 #endif

28 typedef id_t fsh_handle_t;
29 typedef id_t fsh_callback_handle_t;

31 struct fsh_int;
32 typedef struct fsh_int fsh_int_t;

34 typedef struct fsh {
35     void *arg;
36     void (*remove_cb)(void *, fsh_handle_t);

38     /* vnode */
39     int (*read)(fsh_int_t *, void *, vnode_t *, uio_t *, int, cred_t *,
40               caller_context_t *);
41     int (*write)(fsh_int_t *, void *, vnode_t *, uio_t *, int, cred_t *,
42               caller_context_t *);

44     /* vfs */
45     int (*mount)(fsh_int_t *, void *, vfs_t *, vnode_t *, struct mounta *,
46               cred_t *);
47     int (*unmount)(fsh_int_t *, void *, vfs_t *, int, cred_t *);
48 } fsh_t;

50 typedef struct fsh_callback {
51     void *fshc_arg;
52     void (*fshc_free)(vfs_t *, void *);
53     void (*fshc_mount)(vfs_t *, void *);
54 } fsh_callback_t;

56 /* API */
57 extern fsh_handle_t fsh_hook_install(vfs_t *, fsh_t *);
58 extern int fsh_hook_remove(fsh_handle_t);

60 extern fsh_callback_handle_t fsh_callback_install(fsh_callback_t *);
61 extern int fsh_callback_remove(fsh_callback_handle_t);

```

```

63 extern void fsh_fs_enable(vfs_t *);
64 extern void fsh_fs_disable(vfs_t *);

66 /* fsh control passing */
67 extern int fsh_next_read(fsh_int_t *, vnode_t *, uio_t *, int, cred_t *,
68                          caller_context_t *);
69 extern int fsh_next_write(fsh_int_t *, vnode_t *, uio_t *, int, cred_t *,
70                           caller_context_t *);

72 extern int fsh_next_mount(fsh_int_t *, vfs_t *, vnode_t *, struct mounta *uap,
73                           cred_t *);
74 extern int fsh_next_unmount(fsh_int_t *, vfs_t *, int, cred_t *);

76 #ifdef __cplusplus
77 }
78 #endif

80 #endif /* _FSH_H */

```

new/usr/src/uts/common/sys/fsh\_impl.h

1

\*\*\*\*\*

1413 Tue Sep 3 18:09:42 2013

new/usr/src/uts/common/sys/fsh\_impl.h

fsh, fsd, libfsd, fsdadm

\*\*\*\*\*

```
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 Damian Bogel. All rights reserved.
14 */

16 /*
17  * This file includes all the necessary declarations for vfs.c and vnode.c.
18 */

20 #ifndef _FSH_IMPL_H
21 #define _FSH_IMPL_H

23 #include <sys/fsh.h>
24 #include <sys/list.h>
25 #include <sys/pathname.h>
26 #include <sys/types.h>
27 #include <sys/vfs.h>
28 #include <sys/vnode.h>

30 #ifdef __cplusplus
31 extern "C" {
32 #endif

34 struct fsh_fshrecord;
35 typedef struct fsh_fsrecord fsh_fsrecord_t;

37 /* API for vnode.c and vfs.c only */
38 /* vnode.c */
39 extern int fsh_read(vnode_t *, uio_t *, int, cred_t *, caller_context_t *);
40 extern int fsh_write(vnode_t *, uio_t *, int, cred_t *, caller_context_t *);

42 /* vfs.c */
43 extern void fsh_init(void);
44 extern void fsh_exec_mount_callbacks(vfs_t *);
45 extern void fsh_exec_free_callbacks(vfs_t *);
46 extern void fsh_fsrec_destroy(fsh_fsrecord_t *volatile);

48 extern int fsh_mount(vfs_t *, vnode_t *, struct mount *, cred_t *);
49 extern int fsh_unmount(vfs_t *, int, cred_t *);

51 #ifdef __cplusplus
52 }
53 #endif

55 #endif /* _FSH_IMPL_H */
```

```

*****
21197 Tue Sep 3 18:09:42 2013
new/usr/src/uts/common/sys/vfs.h
fsh, fsd, libfsd, fsdadm
*****
_____unchanged_portion_omitted_____

173 extern avl_tree_t      vskstat_tree;
174 extern kmutex_t        vskstat_tree_lock;

176 /*
177 * Structure per mounted file system.  Each mounted file system has
178 * an array of operations and an instance record.
179 *
180 * The file systems are kept on a doubly linked circular list headed by
181 * "rootvfs".
182 * File system implementations should not access this list;
183 * it's intended for use only in the kernel's vfs layer.
184 *
185 * Each zone also has its own list of mounts, containing filesystems mounted
186 * somewhere within the filesystem tree rooted at the zone's rootpath.  The
187 * list is doubly linked to match the global list.
188 *
189 * mnttab locking: the in-kernel mnttab uses the vfs_mntpt, vfs_resource and
190 * vfs_mntopts fields in the vfs_t.  mntpt and resource are refstr_ts that
191 * are set at mount time and can only be modified during a remount.
192 * It is safe to read these fields if you can prevent a remount on the vfs,
193 * or through the convenience funcs vfs_getmntpoint() and vfs_getresource().
194 * The mntopts field may only be accessed through the provided convenience
195 * functions, as it is protected by the vfs list lock.  Modifying a mount
196 * option requires grabbing the vfs list write lock, which can be a very
197 * high latency lock.
198 */
199 struct zone;          /* from zone.h */
200 struct fem_head;     /* from fem.h */
201 struct fsh_fsrecord; /* from fsh_impl.h */

203 typedef struct vfs {
204     struct vfs      *vfs_next;          /* next VFS in VFS list */
205     struct vfs      *vfs_prev;          /* prev VFS in VFS list */
206
207 /* vfs_op should not be used directly.  Accessor functions are provided */
208     vfsops_t        *vfs_op;           /* operations on VFS */
209
210     struct vnode    *vfs_vnodecovered; /* vnode mounted on */
211     uint_t          vfs_flag;           /* flags */
212     uint_t          vfs_bsize;          /* native block size */
213     int             vfs_fstype;         /* file system type index */
214     fsid_t          vfs_fsid;           /* file system id */
215     void            *vfs_data;          /* private data */
216     dev_t           vfs_dev;            /* device of mounted VFS */
217     ulong_t         vfs_bcount;         /* I/O count (accounting) */
218     struct vfs      *vfs_list;          /* sync list pointer */
219     struct vfs      *vfs_hash;          /* hash list pointer */
220     ksema_t         vfs_reflock;        /* mount/unmount/sync lock */
221     uint_t          vfs_count;          /* vfs reference count */
222     mntopts_t       vfs_mntopts;        /* options mounted with */
223     refstr_t        *vfs_resource;      /* mounted resource name */
224     refstr_t        *vfs_mntpt;         /* mount point name */
225     time_t          vfs_mtime;          /* time we were mounted */
226     struct vfs_impl *vfs_implp;         /* impl specific data */
227     /*
228     * Zones support.  Note that the zone that "owns" the mount isn't
229     * necessarily the same as the zone in which the zone is visible.
230     * That is, vfs_zone and (vfs_zone_next|vfs_zone_prev) may refer to
231     * different zones.

```

```

232     /*
233     struct zone      *vfs_zone;          /* zone that owns the mount */
234     struct vfs      *vfs_zone_next;     /* next VFS visible in zone */
235     struct vfs      *vfs_zone_prev;     /* prev VFS visible in zone */
236
237     struct fem_head *vfs_femhead;       /* fs monitoring */
238     minor_t         vfs_lofi_minor;     /* minor if lofi mount */
239
240     struct fsh_fsrecord *volatile
241         vfs_fshrecord;                  /* fs hooking */
242 } vfs_t;
_____unchanged_portion_omitted_____

```

```

*****
16957 Tue Sep 3 18:09:42 2013
new/usr/src/uts/intel/Makefile.intel.shared
fsh, fsd, libfsd, fsdadm
*****
1 # CDDL HEADER START
2 #
3 # The contents of this file are subject to the terms of the
4 # Common Development and Distribution License (the "License").
5 # You may not use this file except in compliance with the License.
6 #
7 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
8 # or http://www.opensolaris.org/os/licensing.
9 # See the License for the specific language governing permissions
10 # and limitations under the License.
11 #
12 # When distributing Covered Code, include this CDDL HEADER in each
13 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
14 # If applicable, add the following below this CDDL HEADER, with the
15 # fields enclosed by brackets "[]" replaced with your own identifying
16 # information: Portions Copyright [yyyy] [name of copyright owner]
17 #
18 # CDDL HEADER END
19 #
21 # Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
22 # Copyright (c) 2012 Nexenta Systems, Inc. All rights reserved.
23 #
24 #
25 # This makefile contains the common definitions for all intel
26 # implementation architecture independent modules.
27 #
28 #
29 #
30 # Machine type (implementation architecture):
31 #
32 PLATFORM = i86pc
33 #
34 #
35 # Everybody needs to know how to build modstubs.o and to locate unix.o.
36 # Note that unix.o must currently be selected from among the possible
37 # "implementation architectures". Note further, that unix.o is only
38 # used as an optional error check for undefines so (theoretically)
39 # any "implementation architectures" could be used. We choose i86pc
40 # because it is the reference port.
41 #
42 UNIX_DIR = $(UTSBASE)/i86pc/unix
43 GENLIB_DIR = $(UTSBASE)/intel/genunix
44 IPDRV_DIR = $(UTSBASE)/intel/ip
45 MODSTUBS_DIR = $(UNIX_DIR)
46 DSF_DIR = $(UTSBASE)/$(PLATFORM)/genassym
47 LINTS_DIR = $(OBJS_DIR)
48 LINT_LIB_DIR = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)
49 #
50 UNIX_O = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
51 GENLIB = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so
52 MODSTUBS_O = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
53 LINT_LIB = $(UTSBASE)/i86pc/lint-libs/$(OBJS_DIR)/llib-lunix.ln
54 GEN_LINT_LIB = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)/llib-lgenunix.ln
55 #
56 #
57 # Include the makefiles which define build rule templates, the
58 # collection of files per module, and a few specific flags. Note
59 # that order is significant, just as with an include path. The
60 # first build rule template which matches the files name will be
61 # used. By including these in order from most machine dependent

```

```

62 # to most machine independent, we allow a machine dependent file
63 # to be used in preference over a machine independent version
64 # (Such as a machine specific optimization, which preserves the
65 # interfaces.)
66 #
67 include $(UTSTREE)/intel/Makefile.files
68 include $(UTSTREE)/common/Makefile.files
69 #
70 #
71 # ----- TRANSITIONAL SECTION -----
72 #
73 #
74 #
75 # Not everything which *should* be a module is a module yet. The
76 # following is a list of such objects which are currently part of
77 # genunix but which might someday become kmods. This must be
78 # defined before we include Makefile.uts, or else genunix's build
79 # won't be as parallel as we might like.
80 #
81 NOT_YET_KMODS = $(OLDPTY_OBJS) $(PTY_OBJS) $(VCONS_CONF_OBJS) $(MOD_OBJS)
82 #
83 #
84 # ----- END OF TRANSITIONAL SECTION -----
85 #
86 # Include machine independent rules. Note that this does not imply
87 # that the resulting module from rules in Makefile.uts is machine
88 # independent. Only that the build rules are machine independent.
89 #
90 include $(UTSBASE)/Makefile.uts
91 #
92 #
93 # The following must be defined for all implementations:
94 #
95 MODSTUBS = $(UTSBASE)/intel/ia32/ml/modstubs.s
96 #
97 #
98 # Define supported builds
99 #
100 DEF_BUILDS = $(DEF_BUILDS64) $(DEF_BUILDS32)
101 ALL_BUILDS = $(ALL_BUILDS64) $(ALL_BUILDS32)
102 #
103 #
104 # x86 or amd64 inline templates
105 #
106 INLINES_32 = $(UTSBASE)/intel/ia32/ml/ia32.il
107 INLINES_64 = $(UTSBASE)/intel/amd64/ml/amd64.il
108 INLINES += $(INLINES_$(CLASS))
109 #
110 #
111 # kernel-specific optimizations; override default in Makefile.master
112 #
113 #
114 CFLAGS_XARCH_32 = $(i386_CFLAGS)
115 CFLAGS_XARCH_64 = $(amd64_CFLAGS)
116 CFLAGS_XARCH = $(CFLAGS_XARCH_$(CLASS))
117 #
118 COPTFLAG_32 = $(COPTFLAG)
119 COPTFLAG_64 = $(COPTFLAG64)
120 COPTIMIZE = $(COPTFLAG_$(CLASS))
121 #
122 CFLAGS = $(CFLAGS_XARCH)
123 CFLAGS += $(COPTIMIZE)
124 CFLAGS += $(INLINES) -D_ASM_INLINES
125 CFLAGS += $(CCMODE)
126 CFLAGS += $(SPACEFLAG)
127 CFLAGS += $(CCUNBOUND)

```

## new/usr/src/uts/intel/Makefile.intel.shared

3

```

128 CFLAGS          += $(CFLAGS_uts)
129 CFLAGS          += -xstrconst

131 ASFLAGS_XARCH_32 = $(i386_ASFLAGS)
132 ASFLAGS_XARCH_64 = $(amd64_ASFLAGS)
133 ASFLAGS_XARCH    = $(ASFLAGS_XARCH_$(CLASS))

135 ASFLAGS          += $(ASFLAGS_XARCH)

137 #
138 #       Define the base directory for installation.
139 #
140 BASE_INS_DIR     = $(ROOT)

142 #
143 #       Debugging level
144 #
145 #       Special knowledge of which special debugging options affect which
146 #       file is used to optimize the build if these flags are changed.
147 #
148 DEBUG_DEFS_OBJ32 =
149 DEBUG_DEFS_DBG32 = -DDEBUG
150 DEBUG_DEFS_OBJ64 =
151 DEBUG_DEFS_DBG64 = -DDEBUG
152 DEBUG_DEFS       = $(DEBUG_DEFS_$(BUILD_TYPE))

154 DEBUG_COND_OBJ32 =:sh = echo \\043
155 DEBUG_COND_DBG32 =
156 DEBUG_COND_OBJ64 =:sh = echo \\043
157 DEBUG_COND_DBG64 =
158 IF_DEBUG_OBJ     = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

160 $(IF_DEBUG_OBJ)syscall.o      :=      DEBUG_DEFS      += -DSYSCALLTRACE
161 $(IF_DEBUG_OBJ)clock.o       :=      DEBUG_DEFS      += -DKSLICE=1

163 #
164 #       Collect the preprocessor definitions to be associated with *all*
165 #       files.
166 #
167 ALL_DEFS          = $(DEBUG_DEFS) $(OPTION_DEFS)

169 #
170 #       The kernels modules which are "implementation architecture"
171 #       specific for this machine are enumerated below. Note that most
172 #       of these modules must exist (in one form or another) for each
173 #       architecture.
174 #
175 #       Common Drivers (usually pseudo drivers) (/kernel/drv)
176 #       DRV_KMODS are built both 32-bit and 64-bit
177 #       DRV_KMODS_32 are built only 32-bit
178 #       DRV_KMODS_64 are built only 64-bit
179 #
180 DRV_KMODS        += aac
181 DRV_KMODS        += aggr
182 DRV_KMODS        += ahci
183 DRV_KMODS        += amd64_gart
184 DRV_KMODS        += amr
185 DRV_KMODS        += apgart
186 DRV_KMODS        += srn
187 DRV_KMODS        += agptarget
188 DRV_KMODS        += arn
189 DRV_KMODS        += arp
190 DRV_KMODS        += asy
191 DRV_KMODS        += ata
192 DRV_KMODS        += ath
193 DRV_KMODS        += atu

```

## new/usr/src/uts/intel/Makefile.intel.shared

4

```

194 DRV_KMODS        += audio
195 DRV_KMODS        += audio1575
196 DRV_KMODS        += audio810
197 DRV_KMODS        += audiocmi
198 DRV_KMODS        += audiocmihd
199 DRV_KMODS        += audioemul0k
200 DRV_KMODS        += audioens
201 DRV_KMODS        += audiohd
202 DRV_KMODS        += audioixp
203 DRV_KMODS        += audiols
204 DRV_KMODS        += audiopl6x
205 DRV_KMODS        += audiopci
206 DRV_KMODS        += audiosolo
207 DRV_KMODS        += audiots
208 DRV_KMODS        += audiovial823x
209 DRV_KMODS_32    += audiovial97
210 DRV_KMODS        += bl
211 DRV_KMODS        += blkdev
212 DRV_KMODS        += bge
213 DRV_KMODS        += bofi
214 DRV_KMODS        += bpf
215 DRV_KMODS        += bridge
216 DRV_KMODS        += bscbus
217 DRV_KMODS        += bscv
218 DRV_KMODS        += chxge
219 DRV_KMODS        += cxgbe
220 DRV_KMODS        += ntxn
221 DRV_KMODS        += myril0ge
222 DRV_KMODS        += clone
223 DRV_KMODS        += cmdk
224 DRV_KMODS        += cn
225 DRV_KMODS        += conskbd
226 DRV_KMODS        += consms
227 DRV_KMODS        += cpuid
228 DRV_KMODS        += cpunex
229 DRV_KMODS        += crypto
230 DRV_KMODS        += cryptoadm
231 DRV_KMODS        += dca
232 DRV_KMODS        += devinfo
233 DRV_KMODS        += dld
234 DRV_KMODS        += dlpiustub
235 DRV_KMODS_32    += dnet
236 DRV_KMODS        += dump
237 DRV_KMODS        += ecpp
238 DRV_KMODS        += emlxs
239 DRV_KMODS        += fd
240 DRV_KMODS        += fdc
241 DRV_KMODS        += fm
242 DRV_KMODS      += fsd
243 DRV_KMODS        += fssnap
244 DRV_KMODS        += hxge
245 DRV_KMODS        += i8042
246 DRV_KMODS        += i915
247 DRV_KMODS        += icmp
248 DRV_KMODS        += icmp6
249 DRV_KMODS        += intel_nb5000
250 DRV_KMODS        += intel_nhm
251 DRV_KMODS        += ip
252 DRV_KMODS        += ip6
253 DRV_KMODS        += ipf
254 DRV_KMODS        += ipnet
255 DRV_KMODS        += ippctl
256 DRV_KMODS        += ipsecah
257 DRV_KMODS        += ipsecesp
258 DRV_KMODS        += ipw
259 DRV_KMODS        += iwh

```

```

260 DRV_KMODS += iwi
261 DRV_KMODS += iwk
262 DRV_KMODS += iwp
263 DRV_KMODS += iwscn
264 DRV_KMODS += kb8042
265 DRV_KMODS += keysock
266 DRV_KMODS += kssl
267 DRV_KMODS += kstat
268 DRV_KMODS += ksyms
269 DRV_KMODS += kmdb
270 DRV_KMODS += llcl
271 DRV_KMODS += lofi
272 DRV_KMODS += log
273 DRV_KMODS += logindmux
274 DRV_KMODS += mega_sas
275 DRV_KMODS += mc-amd
276 DRV_KMODS += mm
277 DRV_KMODS += mouse8042
278 DRV_KMODS += mpt_sas
279 DRV_KMODS += mr_sas
280 DRV_KMODS += mwl
281 DRV_KMODS += nca
282 DRV_KMODS += nsmb
283 DRV_KMODS += nulldriver
284 DRV_KMODS += nv_sata
285 DRV_KMODS += nxge
286 DRV_KMODS += oce
287 DRV_KMODS += openeep
288 DRV_KMODS += pci_pci
289 DRV_KMODS += pcic
290 DRV_KMODS += pcieb
291 DRV_KMODS += phymem
292 DRV_KMODS += pcan
293 DRV_KMODS += pcwl
294 DRV_KMODS += pit_bEEP
295 DRV_KMODS += pm
296 DRV_KMODS += poll
297 DRV_KMODS += pool
298 DRV_KMODS += power
299 DRV_KMODS += pseudo
300 DRV_KMODS += ptc
301 DRV_KMODS += ptm
302 DRV_KMODS += pts
303 DRV_KMODS += ptsl
304 DRV_KMODS += qlge
305 DRV_KMODS += radeon
306 DRV_KMODS += ral
307 DRV_KMODS += ramdisk
308 DRV_KMODS += random
309 DRV_KMODS += rds
310 DRV_KMODS += rdsv3
311 DRV_KMODS += rpcib
312 DRV_KMODS += rsm
313 DRV_KMODS += rts
314 DRV_KMODS += rtw
315 DRV_KMODS += rum
316 DRV_KMODS += rwd
317 DRV_KMODS += rwn
318 DRV_KMODS += sad
319 DRV_KMODS += sd
320 DRV_KMODS += sdhost
321 DRV_KMODS += sgen
322 DRV_KMODS += si3124
323 DRV_KMODS += smbios
324 DRV_KMODS += softmac
325 DRV_KMODS += spdsock

```

```

326 DRV_KMODS += sbsrv
327 DRV_KMODS += smp
328 DRV_KMODS += spps
329 DRV_KMODS += sppptun
330 DRV_KMODS += srpt
331 DRV_KMODS += st
332 DRV_KMODS += sy
333 DRV_KMODS += sysevent
334 DRV_KMODS += sysmsg
335 DRV_KMODS += tcp
336 DRV_KMODS += tcp6
337 DRV_KMODS += tl
338 DRV_KMODS += tnf
339 DRV_KMODS += tpm
340 DRV_KMODS += trill
341 DRV_KMODS += udp
342 DRV_KMODS += udp6
343 DRV_KMODS += ucode
344 DRV_KMODS += ural
345 DRV_KMODS += uath
346 DRV_KMODS += urtw
347 DRV_KMODS += vgatext
348 DRV_KMODS += heci
349 DRV_KMODS += vnic
350 DRV_KMODS += vscan
351 DRV_KMODS += wc
352 DRV_KMODS += winlock
353 DRV_KMODS += wpi
354 DRV_KMODS += xge
355 DRV_KMODS += yge
356 DRV_KMODS += zcons
357 DRV_KMODS += zyd
358 DRV_KMODS += simnet
359 DRV_KMODS += stmf
360 DRV_KMODS += stmf_sbd
361 DRV_KMODS += fct
362 DRV_KMODS += fcoe
363 DRV_KMODS += fcoet
364 DRV_KMODS += fcoei
365 DRV_KMODS += qlt
366 DRV_KMODS += iscsit
367 DRV_KMODS += pppt
368 DRV_KMODS += ncall nsctl sdbc nskern sv
369 DRV_KMODS += ii rdc rdcsrv rdcstub
370 DRV_KMODS += iptun

372 $(CLOSED_BUILD)CLOSED_DRV_KMODS += bmc
373 $(CLOSED_BUILD)CLOSED_DRV_KMODS += glm
374 $(CLOSED_BUILD)CLOSED_DRV_KMODS += intel_nhmex
375 $(CLOSED_BUILD)CLOSED_DRV_KMODS += cpgary3
376 $(CLOSED_BUILD)CLOSED_DRV_KMODS += marvell88sx
377 $(CLOSED_BUILD)CLOSED_DRV_KMODS += bcm_sata
378 $(CLOSED_BUILD)CLOSED_DRV_KMODS += memtest
379 $(CLOSED_BUILD)CLOSED_DRV_KMODS += mpt
380 $(CLOSED_BUILD)CLOSED_DRV_KMODS += atiatom
381 $(CLOSED_BUILD)CLOSED_DRV_KMODS += acpi_toshiba

383 #
384 # Common code drivers
385 #

387 DRV_KMODS += afe
388 DRV_KMODS += atge
389 DRV_KMODS += bfe
390 DRV_KMODS += dmfe
391 DRV_KMODS += e1000g

```

```

392 DRV_KMODS      += efe
393 DRV_KMODS      += elxl
394 DRV_KMODS      += hme
395 DRV_KMODS      += mxfe
396 DRV_KMODS      += nge
397 DRV_KMODS      += pcn
398 DRV_KMODS      += rge
399 DRV_KMODS      += rtls
400 DRV_KMODS      += sfe
401 DRV_KMODS      += amd8111s
402 DRV_KMODS      += igb
403 DRV_KMODS      += ipmi
404 DRV_KMODS      += iprb
405 DRV_KMODS      += ixgbe
406 DRV_KMODS      += vr
407 $(CLOSED_BUILD)CLOSED_DRV_KMODS += ixgb

409 #
410 # Virtio drivers
411 #

413 # Virtio core
414 DRV_KMODS      += virtio

416 # Virtio block driver
417 DRV_KMODS      += vioblk

419 #
420 #      DTrace and DTrace Providers
421 #
422 DRV_KMODS      += dtrace
423 DRV_KMODS      += fbt
424 DRV_KMODS      += lockstat
425 DRV_KMODS      += profile
426 DRV_KMODS      += sdt
427 DRV_KMODS      += systrace
428 DRV_KMODS      += fasttrap
429 DRV_KMODS      += dcpc

431 #
432 #      I/O framework test drivers
433 #
434 DRV_KMODS      += pshot
435 DRV_KMODS      += gen_drv
436 DRV_KMODS      += tvhci tphci tclient
437 DRV_KMODS      += emul64

439 #
440 #      Machine Specific Driver Modules (/kernel/drv):
441 #
442 DRV_KMODS      += options
443 DRV_KMODS      += scsi_vhci
444 DRV_KMODS      += pmcs
445 DRV_KMODS      += pmcs8001fw
446 DRV_KMODS      += arcmsr
447 DRV_KMODS      += fcp
448 DRV_KMODS      += fcip
449 DRV_KMODS      += fcsm
450 DRV_KMODS      += fp
451 DRV_KMODS      += qlc
452 DRV_KMODS      += iscsi

454 #
455 #      PCMCIA specific module(s)
456 #
457 DRV_KMODS      += pcs

```

```

458 DRV_KMODS      += pcata
459 MISC_KMODS     += cardbus
460 $(CLOSED_BUILD)CLOSED_DRV_KMODS += pcser

462 #
463 #      SCSI Enclosure Services driver
464 #
465 DRV_KMODS      += ses

467 #
468 #      USB specific modules
469 #
470 DRV_KMODS      += hid
471 DRV_KMODS      += hwarc hwahc
472 DRV_KMODS      += hubd
473 DRV_KMODS      += uhci
474 DRV_KMODS      += ehci
475 DRV_KMODS      += ohci
476 DRV_KMODS      += usb_mid
477 DRV_KMODS      += usb_ia
478 DRV_KMODS      += scsa2usb
479 DRV_KMODS      += usbprn
480 DRV_KMODS      += ugen
481 DRV_KMODS      += usbser
482 DRV_KMODS      += usbsacm
483 DRV_KMODS      += usbsksp
484 DRV_KMODS      += usbsprl
485 DRV_KMODS      += usb_ac
486 DRV_KMODS      += usb_as
487 DRV_KMODS      += usbskel
488 DRV_KMODS      += usbvc
489 DRV_KMODS      += usbftdi
490 DRV_KMODS      += wusb_df
491 DRV_KMODS      += wusb_ca
492 DRV_KMODS      += usbecm

494 $(CLOSED_BUILD)CLOSED_DRV_KMODS += usbser_edge

496 #
497 #      1394 modules
498 #
499 MISC_KMODS     += s1394 sbp2
500 DRV_KMODS      += hci1394 scsa1394
501 DRV_KMODS      += avl1394
502 DRV_KMODS      += dcam1394

504 #
505 #      InfiniBand pseudo drivers
506 #
507 DRV_KMODS      += ib ibp eibnx eoib rdsib sdp iser daplt hermon tavor sol_ucma
508 DRV_KMODS      += sol_umad

510 #
511 #      LVM modules
512 #
513 DRV_KMODS      += md
514 MISC_KMODS     += md_stripe md_hotspares md_mirror md_raid md_trans md_notify
515 MISC_KMODS     += md_sp

517 #
518 #      Brand modules
519 #
520 BRAND_KMODS    += snl_brand s10_brand

522 #
523 #      Exec Class Modules (/kernel/exec):

```

```

524 #
525 EXEC_KMODS      += elfexec intpexec shbinexec javaexec

527 #
528 #       Scheduling Class Modules (/kernel/sched):
529 #
530 SCHED_KMODS     += IA RT TS RT_DPTBL TS_DPTBL FSS FX FX_DPTBL SDC

532 #
533 #       File System Modules (/kernel/fs):
534 #
535 FS_KMODS        += autofs cacheefs ctfs dcfs dev devfs fdfs fifofs hsfs lofs
536 FS_KMODS        += mntfs namefs nfs objfs zfs zut
537 FS_KMODS        += pcfs procfs sockfs specfs tmpfs udfs ufs sharefs
538 FS_KMODS        += smbfs

540 #
541 #       Streams Modules (/kernel/strmod):
542 #
543 STRMOD_KMODS    += bufmod connld dedump ldterm pckt pfmod pipemod
544 STRMOD_KMODS    += ptem redirmod rpcmod rlmod telmod timod
545 STRMOD_KMODS    += sppsasyn sppscomp
546 STRMOD_KMODS    += tirdwr ttcompat
547 STRMOD_KMODS    += usbkbm
548 STRMOD_KMODS    += usbms
549 STRMOD_KMODS    += usbwcm
550 STRMOD_KMODS    += usb_ah
551 STRMOD_KMODS    += drcompat
552 STRMOD_KMODS    += cryptmod
553 STRMOD_KMODS    += vuid2ps2
554 STRMOD_KMODS    += vuid3ps2
555 STRMOD_KMODS    += vuidm3p
556 STRMOD_KMODS    += vuidm4p
557 STRMOD_KMODS    += vuidm5p

559 #
560 #       'System' Modules (/kernel/sys):
561 #
562 SYS_KMODS       += c2audit
563 SYS_KMODS       += doorfs
564 SYS_KMODS       += exacctsys
565 SYS_KMODS       += inst_sync
566 SYS_KMODS       += kaio
567 SYS_KMODS       += msgsys
568 SYS_KMODS       += pipe
569 SYS_KMODS       += portfs
570 SYS_KMODS       += pset
571 SYS_KMODS       += semsys
572 SYS_KMODS       += shmsys
573 SYS_KMODS       += sysacct
574 SYS_KMODS       += acctctl

576 #
577 #       'Misc' Modules (/kernel/misc)
578 #       MISC_KMODS are built both 32-bit and 64-bit
579 #       MISC_KMODS_32 are built only 32-bit
580 #       MISC_KMODS_64 are built only 64-bit
581 #
582 MISC_KMODS      += ac97
583 MISC_KMODS      += acpica
584 MISC_KMODS      += agpmaster
585 MISC_KMODS      += bignum
586 MISC_KMODS      += bootdev
587 MISC_KMODS      += busra
588 MISC_KMODS      += cmlb
589 MISC_KMODS      += consconfig

```

```

590 MISC_KMODS      += ctf
591 MISC_KMODS      += dadk
592 MISC_KMODS      += dcopy
593 MISC_KMODS      += dls
594 MISC_KMODS      += drm
595 MISC_KMODS      += fssnap_if
596 MISC_KMODS      += gda
597 MISC_KMODS      += gld
598 MISC_KMODS      += hidparser
599 MISC_KMODS      += hook
600 MISC_KMODS      += hpcsvc
601 MISC_KMODS      += ibcm
602 MISC_KMODS      += ibdm
603 MISC_KMODS      += ibdma
604 MISC_KMODS      += ibmf
605 MISC_KMODS      += ibtl
606 MISC_KMODS      += idm
607 MISC_KMODS      += idmap
608 MISC_KMODS      += iomulib
609 MISC_KMODS      += ipc
610 MISC_KMODS      += kbtrans
611 MISC_KMODS      += kcf
612 MISC_KMODS      += kgssapi
613 MISC_KMODS      += kmecch_dummy
614 MISC_KMODS      += kmecch_krb5
615 MISC_KMODS      += ksocket
616 MISC_KMODS      += mac
617 MISC_KMODS      += mii
618 MISC_KMODS      += mwlw
619 MISC_KMODS      += net80211
620 MISC_KMODS      += nfs_dlboot
621 MISC_KMODS      += nfssrv
622 MISC_KMODS      += neti
623 MISC_KMODS      += pci_autoconfig
624 MISC_KMODS      += pcicfg
625 MISC_KMODS      += pcihp
626 MISC_KMODS      += pcmcia
627 MISC_KMODS      += rpcsec
628 MISC_KMODS      += rpcsec_gss
629 MISC_KMODS      += rsmops
630 MISC_KMODS      += sata
631 MISC_KMODS      += scsi
632 MISC_KMODS      += sda
633 MISC_KMODS      += sol_ofs
634 MISC_KMODS      += spuni
635 MISC_KMODS      += strategy
636 MISC_KMODS      += strplumb
637 MISC_KMODS      += tem
638 MISC_KMODS      += tlimod
639 MISC_KMODS      += usba usba10 usbs49_fw
640 MISC_KMODS      += scsi_vhci_f_sym_hds
641 MISC_KMODS      += scsi_vhci_f_sym
642 MISC_KMODS      += scsi_vhci_f_tpgs
643 MISC_KMODS      += scsi_vhci_f_asym_sun
644 MISC_KMODS      += scsi_vhci_f_tape
645 MISC_KMODS      += scsi_vhci_f_tpgs_tape
646 MISC_KMODS      += fctl
647 MISC_KMODS      += emlxs_fw
648 MISC_KMODS      += qlc_fw_2200
649 MISC_KMODS      += qlc_fw_2300
650 MISC_KMODS      += qlc_fw_2400
651 MISC_KMODS      += qlc_fw_2500
652 MISC_KMODS      += qlc_fw_6322
653 MISC_KMODS      += qlc_fw_8100
654 MISC_KMODS      += hwal480_fw
655 MISC_KMODS      += uathfw

```



```

656 MISC_KMODS      += uwba

658 $(CLOSED_BUILD)CLOSED_MISC_KMODS      += klmmod klmops
659 $(CLOSED_BUILD)CLOSED_MISC_KMODS      += scsi_vhci_f_asym_lsi
660 $(CLOSED_BUILD)CLOSED_MISC_KMODS      += scsi_vhci_f_asym_emc
661 $(CLOSED_BUILD)CLOSED_MISC_KMODS      += scsi_vhci_f_sym_emc

663 #
664 #       Software Cryptographic Providers (/kernel/crypto):
665 #
666 CRYPTO_KMODS    += aes
667 CRYPTO_KMODS    += arcfour
668 CRYPTO_KMODS    += blowfish
669 CRYPTO_KMODS    += des
670 CRYPTO_KMODS    += ecc
671 CRYPTO_KMODS    += md4
672 CRYPTO_KMODS    += md5
673 CRYPTO_KMODS    += rsa
674 CRYPTO_KMODS    += sha1
675 CRYPTO_KMODS    += sha2
676 CRYPTO_KMODS    += swrand

678 #
679 #       IP Policy Modules (/kernel/ipp)
680 #
681 IPP_KMODS        += dlcosmk
682 IPP_KMODS        += flowacct
683 IPP_KMODS        += ipgpc
684 IPP_KMODS        += dscpmk
685 IPP_KMODS        += tokenmt
686 IPP_KMODS        += tswtclmt

688 #
689 #       generic-unix module (/kernel/genunix):
690 #
691 GENUNIX_KMODS    += genunix

693 #
694 #       SVVS Testing Modules (/kernel/strmod):
695 #
696 #       These are streams and driver modules which are not to be
697 #       delivered with a released system. However, during development
698 #       it is convenient to build and install the SVVS kernel modules.
699 #
700 SVVS_KMODS       += lmodb lmode lmodr lmodt svvslo tidg tivc tmux

702 $(CLOSED_BUILD)SVVS      += svvs

704 #
705 #       Modules eXcluded from the product:
706 #
707 $(CLOSED_BUILD)CLOSED_XMODS =          \
708     adpu320      \
709     bnx          \
710     bnxe        \
711     lsimega     \
712     sdpib

715 #
716 #       'Dacf' Modules (/kernel/dacf):
717 #

719 #
720 #       Performance Counter BackEnd modules (/usr/kernel/pcbe)
721 #

```

```

722 PCBE_KMODS      += pl23_pcbe p4_pcbe opteron_pcbe core_pcbe

724 #
725 #       MAC-Type Plugin Modules (/kernel/mac)
726 #
727 MAC_KMODS        += mac_6to4
728 MAC_KMODS        += mac_ether
729 MAC_KMODS        += mac_ipv4
730 MAC_KMODS        += mac_ipv6
731 MAC_KMODS        += mac_wifi
732 MAC_KMODS        += mac_ib

734 #
735 #       socketmod (kernel/socketmod)
736 #
737 SOCKET_KMODS     += sockpfp
738 SOCKET_KMODS     += socksctp
739 SOCKET_KMODS     += socksdp
740 SOCKET_KMODS     += sockrds
741 SOCKET_KMODS     += ksslif

743 #
744 #       kiconv modules (/kernel/kiconv):
745 #
746 KICONV_KMODS     += kiconv_emea kiconv_ja kiconv_ko kiconv_sc kiconv_tc

748 #
749 #       'Dacf' Modules (/kernel/dacf):
750 #
751 DACF_KMODS       += net_dacf

```

```
*****
1586 Tue Sep 3 18:09:42 2013
new/usr/src/uts/intel/fsd/Makefile
fsh, fsd, libfsd, fsdadm
*****
```

```
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 2012 Joyent, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
26 #
27 # Copyright 2013 Damian Bogel. All rights reserved.
28 #

30 UTSBASE = ../../

32 MODULE = fsd
33 OBJECTS = $(FSD_OBJS:%=$(OBJS_DIR)/%)
34 LINTS = $(FSD_OBJS:%.o=$(LINTS_DIR)/%.ln)
35 ROOTMODULE = $(ROOT_DRV_DIR)/$(MODULE)
36 CONF_SRCDIR = $(UTSBASE)/common/io/fsd

38 include $(UTSBASE)/intel/Makefile.intel

40 ALL_TARGET = $(BINARY) $(SRC_CONFILE)
41 LINT_TARGET = $(MODULE).lint
42 INSTALL_TARGET = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

44 LDFLAGS += -dy

46 .KEEP_STATE:

48 def: $(DEF_DEPS)

50 all: $(ALL_DEPS)

52 clean: $(CLEAN_DEPS)

54 clobber: $(CLOBBER_DEPS)

56 lint: $(LINT_DEPS)

58 modlintlib: $(MODLINTLIB_DEPS)

60 clean.lint: $(CLEAN_LINT_DEPS)
```

```
62 install: $(INSTALL_DEPS)

64 include $(UTSBASE)/intel/Makefile.targ
```