```
**********************************************************
    7361 Wed Oct 16 17:41:04 2013
new/usr/src/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
  24 # Copyright (c) 2012 by Delphix. All rights reserved.
  25 #

  27 #
  28 # Makefile for system source
  29 #
  30 # include global definitions
  31 include Makefile.master
  32 #
  33 # the Targetdirs file is the AT&T target.dirs file in a makefile format.
  34 # it defines TARGETDIRS and ROOTDIRS.
  35 include Targetdirs

  37 COMMON_SUBDIRS= uts lib cmd ucblib ucbcmd psm man test
  38 sparc_SUBDIRS= stand
  39 i386_SUBDIRS= grub

  41 #
  42 # sparc needs to build stand before psm
  43 #
  44 $(SPARC_BLD)psm: stand

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

  48 HDRSUBDIRS=     uts head lib cmd

  50 # UCB headers are bug-for-bug compatible and not checkable against the header
  51 # standards.
  52 #
  53 CHKHDRSUBDIRS=  head uts lib

  55 #
  56 # Headers that can be built in parallel
  57 #
  58 PARALLEL_HEADERS = sysheaders userheaders libheaders cmdheaders
```



```
  60 #
  61 # Directories that can be built in parallel
  62 #
  63 PARALLEL_DIRS = uts lib man

  65 # The check target also causes smf(5) service manifests to be validated.
  66 CHKMFSTSUBDIRS= cmd

  68 MSGSUBDIRS=     cmd ucbcmd lib
  69 DOMAINS= \
  70         SUNW_OST_ADMIN \
  71         SUNW_OST_NETRPC \
  72         SUNW_OST_OSCMD \
  73         SUNW_OST_OSLIB \
  74         SUNW_OST_UCBCMD \
  75         SUNW_OST_ZONEINFO

  77 MSGDDIRS=       $(DOMAINS:%=$(MSGROOT)/%)
  78 MSGDIRS=        $(MSGROOT) $(MSGDDIRS) $(MSGROOT)/LC_TIME

  80 all :=                  TARGET= all
  81 install :=              TARGET= install
  82 install1 :=             TARGET= install
  83 install2 :=             TARGET= install
  84 install_h :=            TARGET= install_h
  85 clean :=                TARGET= clean
  86 clobber :=              TARGET= clobber
  87 check :=                TARGET= check

  89 .KEEP_STATE:

  91 #
  92 # Note: install does not cause a build in pkg.  To build packages,
  93 #       cd pkg and do a 'make install'
  94 #

  96 all: mapfiles closedbins sgs .WAIT $(SUBDIRS) pkg

  98 #
  99 # The _msg build is a two-step process.  First, the _msg dependency
 100 # causes recursive makes in $(MSGSUBDIRS), which stages raw message
 101 # files in $(ROOT)/catalog.  Second, the action from the install
 102 # target rule causes those messages to be post-processed from where
 103 # they were staged in $(ROOT)/catalog, and the results placed into the
 104 # proto area.
 105 #
 106 # The stage-licenses target causes the license files needed for
 107 # packaging to be pulled from $(SRC) and $(CLOSED) and staged in
 108 # $(ROOT)/licenses.
 109 #
 110 install: install1 install2 _msg stage-licenses
 111         @cd msg; pwd; $(MAKE) _msg
 112         @rm -rf "$(ROOT)/catalog"

 114 stage-licenses: install2
 115         @cd pkg; pwd; $(MAKE) stage-licenses

 117 install1: mapfiles closedbins sgs

 119 install2: install1 $(SUBDIRS)

 121 _msg: _msgdirs rootdirs FRC
 122         @for m in $(MSGSUBDIRS); do \
 123                 cd $$m; pwd; $(MAKE) _msg; cd ..; \
 124         done
```

```
126 mapfiles: bldtools
127         @cd common/mapfiles; pwd; $(MAKE) install

129 clean: $(SUBDIRS) head pkg
130 clobber: $(SUBDIRS) head pkg clobber_local
131 clobber_local:
132         @cd tools; pwd; $(MAKE) clobber
133         @cd common/mapfiles; pwd; $(MAKE) clobber
134         @cd msg; pwd; $(MAKE) clobber

136 closedbins: bldtools $(ROOTDIRS) FRC
137         @CLOSED_ROOT="$$ON_CLOSED_BINS/root_$(MACH)$${RELEASE_BUILD+-nd}"; \
138         if [ "$$CLOSED_IS_PRESENT" = no ]; then \
138         if [ ! -d "$$CLOSED_ROOT" ]; then \
139                 $(ECHO) "Error: ON_CLOSED_BINS must point to closed" \
140                     "binaries."; \
140                         $(ECHO) "Error: if closed sources are not present," \
141                             "ON_CLOSED_BINS must point to closed binaries."; \
141                 $(ECHO) "root_$(MACH)$${RELEASE_BUILD+-nd} is not" \
142                     "present in $$ON_CLOSED_BINS."; \
143                 exit 1; \
144         fi; \
145         $(ECHO) "Copying closed binaries from $$CLOSED_ROOT"; \
146         (cd $$CLOSED_ROOT; \
147             $(TAR) cfX - $(CODEMGR_WS)/exception_lists/closed-bins .) | \
148             (cd $(ROOT); $(TAR) xBpf -); \
149         ( cd $(ROOT); $(CTFSTRIP) $$(cd $$CLOSED_ROOT; $(FIND) \
150             ./kernel ./usr/kernel ./platform/*/kernel -type f -a -perm -u+x | \
151             $(EGREP) -vf $(CODEMGR_WS)/exception_lists/closed-bins) )
152                 $(EGREP) -vf $(CODEMGR_WS)/exception_lists/closed-bins) ); \
153         fi

153 #
154 # Declare what parts can be built in parallel
155 # DUMMY at the end is used in case macro expansion produces an empty string to
156 # prevent everything going in parallel
157 #
158 .PARALLEL: $(PARALLEL_HEADERS) DUMMY
159 .PARALLEL: $(PARALLEL_DIRS) DUMMY

161 $(SUBDIRS) head pkg: FRC
162         @cd $@; pwd; $(MAKE) $(TARGET)

164 # librpcsvc has a dependency on headers installed by
165 # userheaders, hence the .WAIT before libheaders.
166 sgs: rootdirs .WAIT sysheaders userheaders .WAIT \
167         libheaders cmdheaders

169 #
170 # Top-level setup target to setup the development environment that includes
171 # headers, tools and generated mapfiles.  For open-only builds (i.e.: source
172 # trees w/o usr/closed), this also depends on the closedbins target (above)
173 # in order to properly seed the proto area.  Note, although the tools are
174 # dependent on a number of constant mapfiles, the tools themselves are
175 # required to build the generated mapfiles.
176 #
177 setup: closedbins bldtools sgs mapfiles

179 bldtools:
180         @cd tools; pwd; $(MAKE) install

182 # /var/mail/:saved is a special case because of the colon in the name.
183 #
184 rootdirs: $(ROOTDIRS)
185         $(INS) -d -m 775 $(ROOT)/var/mail/:saved
```



```
187 lint: FRC
188         $(MAKE) -f Makefile.lint

190 _msgdirs:       $(MSGDIRS)

192 $(ROOTDIRS) $(MSGDIRS):
193         $(INS.dir)

195 userheaders: FRC
196         @cd head; pwd; $(MAKE) install_h

198 libheaders: bldtools
199         @cd lib; pwd; $(MAKE) install_h

201 sysheaders: FRC
202         @cd uts; pwd; $(MAKE) install_h

204 cmdheaders: FRC
205         @cd cmd/fm; pwd; $(MAKE) install_h
206         @cd cmd/mdb; pwd; $(MAKE) install_h

208 check:  $(CHKHDRSUBDIRS) $(CHKMFSTSUBDIRS)

210 #
211 # Cross-reference customization: skip all of the subdirectories that
212 # don't contain actual source code.
213 #
216 $(CLOSED_BUILD)XRDIRS += ../closed
214 XRPRUNE = pkg prototypes
215 XRINCDIRS = uts/common head ucbhead
219 $(CLOSED_BUILD)XRINCDIRS = uts/common ../closed/uts/common head ucbhead

217 cscope.out tags: FRC
218         $(XREF) -f -x $@

220 FRC:

222 #
223 # Targets for reporting compiler versions; nightly uses these.
224 #

226 cc-version:
227         @if $($(MACH)_CC) -_versions >/dev/null 2>/dev/null; then \
228                 $(ECHO) 32-bit compiler;                          \
229                 $(ECHO) $($(MACH)_CC);                            \
230                 $($(MACH)_CC) -_versions 2>&1 |                   \
231                     $(EGREP) '^(cw|cc|gcc|primary|shadow)';       \
232         else                                                      \
233                 __COMPILER='$($(MACH)_CC) -_compiler 2>/dev/null || $(TRUE)';\
234                 if [ -z "$$__COMPILER" ]; then                    \
235                         $(ECHO) No 32-bit compiler found;         \
236                         exit 1;                                   \
237                 else                                              \
238                         $(ECHO) 32-bit compiler;                  \
239                         $(ECHO) $($(MACH)_CC);                    \
240                         $(ECHO) $$__COMPILER;                     \
241                         $($(MACH)_CC) -V 2>&1 | head -1;          \
242                 fi;                                               \
243         fi

245 cc64-version:
246         @if $($(MACH64)_CC) -_versions >/dev/null 2>/dev/null; then \
247                 $(ECHO) 64-bit compiler;                          \
248                 $(ECHO) $($(MACH64)_CC);                          \
249                 $($(MACH64)_CC) -_versions 2>&1 |                 \
```

```
250                     $(EGREP) '^(cw|cc|gcc|primary|shadow)';     \
251         else                                                    \
252                 __COMPILER=`$($(MACH64)_CC) -_compiler 2>/dev/null || $(TRUE)`;\
253                 if [ -z "$$__COMPILER" ]; then                  \
254                         $(ECHO) No 64-bit compiler found;       \
255                         exit 1;                                 \
256                 else                                            \
257                         $(ECHO) 64-bit compiler;                \
258                         $(ECHO) $($(MACH64)_CC);                \
259                         $(ECHO) $$__COMPILER;                   \
260                         $($(MACH64)_CC) -V 2>&1 | head -1;      \
261                 fi;                                             \
262         fi

264 java-version:
265         @if [ -x "$(JAVAC)" ]; then                     \
266                 $(ECHO) $(JAVAC);                       \
267                 $(JAVA_ROOT)/bin/java -fullversion 2>&1 | head -1;      \
268         else                                            \
269                 $(ECHO) No Java compiler found;         \
270                 exit 1;                                 \
271         fi
```

```
**********************************************************
   35033 Wed Oct 16 17:41:04 2013
new/usr/src/Makefile.master
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2012 by Delphix. All rights reserved.
#

#
# Makefile.master, global definitions for system source
#
ROOT=           /proto

#
# Adjunct root, containing an additional proto area to be used for headers
# and libraries.
#
ADJUNCT_PROTO=

#
# Adjunct for building things that run on the build machine.
#
NATIVE_ADJUNCT= /usr

#
# RELEASE_BUILD should be cleared for final release builds.
# NOT_RELEASE_BUILD is exactly what the name implies.
#
# INTERNAL_RELEASE_BUILD is a subset of RELEASE_BUILD. It mostly controls
# identification strings. Enabling RELEASE_BUILD automatically enables
# INTERNAL_RELEASE_BUILD.
#
# CLOSED_BUILD controls whether we try to build files under
# usr/closed.  ("" means to build closed code, "#" means don't try to
# build it.)  Skipping the closed code implies doing an export release
# build.
#
# STRIP_COMMENTS toggles comment section striping. Generally the same setting
# as INTERNAL_RELEASE_BUILD.
#
```



```
# __GNUC toggles the building of ON components using gcc and related tools.
# Normally set to '#', set it to '' to do gcc build.
#
# The declaration POUND_SIGN is always '#'. This is needed to get around the
# make feature that '#' is always a comment delimiter, even when escaped or
# quoted. We use this macro expansion method to get POUND_SIGN rather than
# always breaking out a shell because the general case can cause a noticable
# slowdown in build times when so many Makefiles include Makefile.master.
#
# While the majority of users are expected to override the setting below
# with an env file (via nightly or bldenv), if you aren't building that way
# (ie, you're using "ws" or some other bootstrapping method) then you need
# this definition in order to avoid the subshell invocation mentioned above.
#

PRE_POUND=                              pre\#
POUND_SIGN=                             $(PRE_POUND:pre\%=%)

NOT_RELEASE_BUILD=
INTERNAL_RELEASE_BUILD=                 $(POUND_SIGN)
RELEASE_BUILD=                          $(POUND_SIGN)
$(RELEASE_BUILD)NOT_RELEASE_BUILD=      $(POUND_SIGN)
$(RELEASE_BUILD)INTERNAL_RELEASE_BUILD=
PATCH_BUILD=                            $(POUND_SIGN)

# If CLOSED_IS_PRESENT is not set, assume the closed tree is present.
CLOSED_BUILD_1= $(CLOSED_IS_PRESENT:yes=)
CLOSED_BUILD=   $(CLOSED_BUILD_1:no=$(POUND_SIGN))

# SPARC_BLD is '#' for an Intel build.
# INTEL_BLD is '#' for a Sparc build.
SPARC_BLD_1=    $(MACH:i386=$(POUND_SIGN))
SPARC_BLD=      $(SPARC_BLD_1:sparc=)
INTEL_BLD_1=    $(MACH:sparc=$(POUND_SIGN))
INTEL_BLD=      $(INTEL_BLD_1:i386=)

STRIP_COMMENTS= $(INTERNAL_RELEASE_BUILD)

# Are we building tonic closedbins? Unless you have used the
# -O flag to nightly or bldenv, leave the definition of TONICBUILD
# as $(POUND_SIGN).
#
# IF YOU CHANGE CLOSEDROOT, you MUST change install.bin
# to match the new definition.
TONICBUILD=     $(POUND_SIGN)
$(TONICBUILD)CLOSEDROOT= $(ROOT)-closed


# The variables below control the compilers used during the build.
# There are a number of permutations.
#
# __GNUC and __SUNC control (and indicate) the primary compiler.  Whichever
# one is not POUND_SIGN is the primary, with the other as the shadow.  They
# may also be used to control entirely compiler-specific Makefile assignments.
# __SUNC and Sun Studio are the default.
#
# __GNUC64 indicates that the 64bit build should use the GNU C compiler.
# There is no Sun C analogue.
#
# The following version-specific options are operative regardless of which
# compiler is primary, and control the versions of the given compilers to be
# used.  They also allow compiler-version specific Makefile fragments.
#

__GNUC=                 $(POUND_SIGN)
$(__GNUC)__SUNC=        $(POUND_SIGN)
```

```
__GNUC64=               $(__GNUC)

# CLOSED is the root of the tree that contains source which isn't released
# as open source
CLOSED=         $(SRC)/../closed

# BUILD_TOOLS is the root of all tools including compilers.
# ONBLD_TOOLS is the root of all the tools that are part of SUNWonbld.

BUILD_TOOLS=            /ws/onnv-tools
ONBLD_TOOLS=            $(BUILD_TOOLS)/onbld

JAVA_ROOT=      /usr/java

SFW_ROOT=       /usr/sfw
SFWINCDIR=      $(SFW_ROOT)/include
SFWLIBDIR=      $(SFW_ROOT)/lib
SFWLIBDIR64=    $(SFW_ROOT)/lib/$(MACH64)

GCC_ROOT=       /opt/gcc/4.4.4
GCCLIBDIR=      $(GCC_ROOT)/lib
GCCLIBDIR64=    $(GCC_ROOT)/lib/$(MACH64)

DOCBOOK_XSL_ROOT=       /usr/share/sgml/docbook/xsl-stylesheets

RPCGEN=         /usr/bin/rpcgen
STABS=          $(ONBLD_TOOLS)/bin/$(MACH)/stabs
ELFEXTRACT=     $(ONBLD_TOOLS)/bin/$(MACH)/elfextract
MBH_PATCH=      $(ONBLD_TOOLS)/bin/$(MACH)/mbh_patch
ECHO=           echo
INS=            install
TRUE=           true
SYMLINK=        /usr/bin/ln -s
LN=             /usr/bin/ln
CHMOD=          /usr/bin/chmod
MV=             /usr/bin/mv -f
RM=             /usr/bin/rm -f
CUT=            /usr/bin/cut
NM=             /usr/ccs/bin/nm
DIFF=           /usr/bin/diff
GREP=           /usr/bin/grep
EGREP=          /usr/bin/egrep
ELFWRAP=        /usr/bin/elfwrap
KSH93=          /usr/bin/ksh93
SED=            /usr/bin/sed
NAWK=           /usr/bin/nawk
CP=             /usr/bin/cp -f
MCS=            /usr/ccs/bin/mcs
CAT=            /usr/bin/cat
ELFDUMP=        /usr/ccs/bin/elfdump
M4=             /usr/ccs/bin/m4
STRIP=          /usr/ccs/bin/strip
LEX=            /usr/ccs/bin/lex
FLEX=           $(SFW_ROOT)/bin/flex
YACC=           /usr/ccs/bin/yacc
CPP=            /usr/lib/cpp
JAVAC=          $(JAVA_ROOT)/bin/javac
JAVAH=          $(JAVA_ROOT)/bin/javah
JAVADOC=        $(JAVA_ROOT)/bin/javadoc
RMIC=           $(JAVA_ROOT)/bin/rmic
JAR=            $(JAVA_ROOT)/bin/jar
CTFCONVERT=     $(ONBLD_TOOLS)/bin/$(MACH)/ctfconvert
CTFMERGE=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfmerge
CTFSTABS=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfstabs
CTFSTRIP=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfstrip
NDRGEN=         $(ONBLD_TOOLS)/bin/$(MACH)/ndrgen
```



```
GENOFFSETS=     $(ONBLD_TOOLS)/bin/genoffsets
CTFCVTPTBL=     $(ONBLD_TOOLS)/bin/ctfcvtptbl
CTFFINDMOD=     $(ONBLD_TOOLS)/bin/ctffindmod
XREF=           $(ONBLD_TOOLS)/bin/xref
FIND=           /usr/bin/find
PERL=           /usr/bin/perl
PYTHON_26=      /usr/bin/python2.6
PYTHON=         $(PYTHON_26)
SORT=           /usr/bin/sort
TOUCH=          /usr/bin/touch
WC=             /usr/bin/wc
XARGS=          /usr/bin/xargs
ELFEDIT=        /usr/bin/elfedit
ELFSIGN=        /usr/bin/elfsign
DTRACE=         /usr/sbin/dtrace -xnolibs
UNIQ=           /usr/bin/uniq
TAR=            /usr/bin/tar

FILEMODE=       644
DIRMODE=        755

#
# The version of the patch makeup table optimized for build-time use.  Used
# during patch builds only.
$(PATCH_BUILD)PMTMO_FILE=$(SRC)/patch_makeup_table.mo

# Declare that nothing should be built in parallel.
# Individual Makefiles can use the .PARALLEL target to declare otherwise.
.NO_PARALLEL:

# For stylistic checks
#
# Note that the X and C checks are not used at this time and may need
# modification when they are actually used.
#
CSTYLE=         $(ONBLD_TOOLS)/bin/cstyle
CSTYLE_TAIL=
HDRCHK=         $(ONBLD_TOOLS)/bin/hdrchk
HDRCHK_TAIL=
JSTYLE=         $(ONBLD_TOOLS)/bin/jstyle

DOT_H_CHECK=    \
        @$(ECHO) "checking $<"; $(CSTYLE) $< $(CSTYLE_TAIL); \
        $(HDRCHK) $< $(HDRCHK_TAIL)

DOT_X_CHECK=    \
        @$(ECHO) "checking $<"; $(RPCGEN) -C -h $< | $(CSTYLE) $(CSTYLE_TAIL); \
        $(RPCGEN) -C -h $< | $(HDRCHK) $< $(HDRCHK_TAIL)

DOT_C_CHECK=    \
        @$(ECHO) "checking $<"; $(CSTYLE) $< $(CSTYLE_TAIL)

MANIFEST_CHECK= \
        @$(ECHO) "checking $<"; \
        SVCCFG_DTD=$(SRC)/cmd/svc/dtd/service_bundle.dtd.1 \
        SVCCFG_REPOSITORY=$(SRC)/cmd/svc/seed/global.db \
        SVCCFG_CONFIGD_PATH=$(SRC)/cmd/svc/configd/svc.configd-native \
        $(SRC)/cmd/svc/svccfg/svccfg-native validate $<

#
# IMPORTANT:: If you change any of INS.file, INS.dir, INS.rename,
# INS.link or INS.symlink here, then you must also change the
# corresponding override definitions in $CLOSED/Makefile.tonic.
# If you do not do this, then the closedbins build for the OpenSolaris
# community will break. PS, the gatekeepers will be upset too.
INS.file=       $(RM) $@; $(INS) -s -m $(FILEMODE) -f $(@D) $<
```

```
INS.dir=        $(INS) -s -d -m $(DIRMODE) $@
# installs and renames at once
#
INS.rename=     $(INS.file); $(MV) $(@D)/$(<F) $@

# install a link
INSLINKTARGET=  $<
INS.link=       $(RM) $@; $(LN) $(INSLINKTARGET) $@
INS.symlink=    $(RM) $@; $(SYMLINK) $(INSLINKTARGET) $@

#
# Python bakes the mtime of the .py file into the compiled .pyc and
# rebuilds if the baked-in mtime != the mtime of the source file
# (rather than only if it's less than), thus when installing python
# files we must make certain to not adjust the mtime of the source
# (.py) file.
#
INS.pyfile=     $(INS.file); $(TOUCH) -r $< $@

# MACH must be set in the shell environment per uname -p on the build host
# More specific architecture variables should be set in lower makefiles.
#
# MACH64 is derived from MACH, and BUILD64 is set to '#' for
# architectures on which we do not build 64-bit versions.
# (There are no such architectures at the moment.)
#
# Set BUILD64=# in the environment to disable 64-bit amd64
# builds on i386 machines.

MACH64_1=       $(MACH:sparc=sparcv9)
MACH64=         $(MACH64_1:i386=amd64)

MACH32_1=       $(MACH:sparc=sparcv7)
MACH32=         $(MACH32_1:i386=i86)

sparc_BUILD64=
i386_BUILD64=
BUILD64=        $($(MACH)_BUILD64)

#
# C compiler mode. Future compilers may change the default on us,
# so force extended ANSI mode globally. Lower level makefiles can
# override this by setting CCMODE.
#
CCMODE=                 -Xa
CCMODE64=               -Xa

#
# C compiler verbose mode. This is so we can enable it globally,
# but turn it off in the lower level makefiles of things we cannot
# (or aren't going to) fix.
#
CCVERBOSE=              -v

# set this to the secret flag "-Wc,-Qiselect-v9abiwarn=1" to get warnings
# from the compiler about places the -xarch=v9 may differ from -xarch=v9c.
V9ABIWARN=

# set this to the secret flag "-Wc,-Qiselect-regsym=0" to disable register
# symbols (used to detect conflicts between objects that use global registers)
# we disable this now for safety, and because genunix doesn't link with
# this feature (the v9 default) enabled.
#
# REGSYM is separate since the C++ driver syntax is different.
CCREGSYM=               -Wc,-Qiselect-regsym=0
CCCREGSYM=              -Qoption cg -Qiselect-regsym=0
```



```
# Prevent the removal of static symbols by the SPARC code generator (cg).
# The x86 code generator (ube) does not remove such symbols and as such
# using this workaround is not applicable for x86.
#
CCSTATICSYM=            -Wc,-Qassembler-ounrefsym=0
#
# generate 32-bit addresses in the v9 kernel. Saves memory.
CCABS32=                -Wc,-xcode=abs32
#
# generate v9 code which tolerates callers using the v7 ABI, for the sake of
# system calls.
CC32BITCALLERS=         -_gcc=-massume-32bit-callers

# GCC, especially, is increasingly beginning to auto-inline functions and
# sadly does so separately not under the general -fno-inline-functions
# Additionally, we wish to prevent optimisations which cause GCC to clone
# functions -- in particular, these may cause unhelpful symbols to be
# emitted instead of function names
CCNOAUTOINLINE= -_gcc=-fno-inline-small-functions \
        -_gcc=-fno-inline-functions-called-once \
        -_gcc=-fno-ipa-cp

# One optimization the compiler might perform is to turn this:
#       #pragma weak foo
#       extern int foo;
#       if (&foo)
#               foo = 5;
# into
#       foo = 5;
# Since we do some of this (foo might be referenced in common kernel code
# but provided only for some cpu modules or platforms), we disable this
# optimization.
#
sparc_CCUNBOUND = -Wd,-xsafe=unboundsym
i386_CCUNBOUND  =
CCUNBOUND       = $($(MACH)_CCUNBOUND)

#
# compiler '-xarch' flag. This is here to centralize it and make it
# overridable for testing.
sparc_XARCH=    -m32
sparcv9_XARCH=  -m64
i386_XARCH=
amd64_XARCH=    -m64 -Ui386 -U__i386

# assembler '-xarch' flag.  Different from compiler '-xarch' flag.
sparc_AS_XARCH=         -xarch=v8plus
sparcv9_AS_XARCH=       -xarch=v9
i386_AS_XARCH=
amd64_AS_XARCH=         -xarch=amd64 -P -Ui386 -U__i386

#
# These flags define what we need to be 'standalone' i.e. -not- part
# of the rather more cosy userland environment.  This basically means
# the kernel.
#
# XX64  future versions of gcc will make -mcmodel=kernel imply -mno-red-zone
#
sparc_STAND_FLAGS=      -_gcc=-ffreestanding
sparcv9_STAND_FLAGS=    -_gcc=-ffreestanding
# Disabling MMX also disables 3DNow, disabling SSE also disables all later
# additions to SSE (SSE2, AVX ,etc.)
NO_SIMD=                -_gcc=-mno-mmx -_gcc=-mno-sse
i386_STAND_FLAGS=       -_gcc=-ffreestanding $(NO_SIMD)
amd64_STAND_FLAGS=      -xmodel=kernel $(NO_SIMD)
```

```
365 SAVEARGS=               -Wu,-save_args
366 amd64_STAND_FLAGS       += $(SAVEARGS)

368 STAND_FLAGS_32 = $($(MACH)_STAND_FLAGS)
369 STAND_FLAGS_64 = $($(MACH64)_STAND_FLAGS)

371 #
372 # disable the incremental linker
373 ILDOFF=                 -xildoff
374 #
375 XDEPEND=                -xdepend
376 XFFLAG=                 -xF=%all
377 XESS=                   -xs
378 XSTRCONST=              -xstrconst

380 #
381 # turn warnings into errors (C)
382 CERRWARN = -errtags=yes -errwarn=%all
383 CERRWARN += -erroff=E_EMPTY_TRANSLATION_UNIT
384 CERRWARN += -erroff=E_STATEMENT_NOT_REACHED

386 CERRWARN += -_gcc=-Wno-missing-braces
387 CERRWARN += -_gcc=-Wno-sign-compare
388 CERRWARN += -_gcc=-Wno-unknown-pragmas
389 CERRWARN += -_gcc=-Wno-unused-parameter
390 CERRWARN += -_gcc=-Wno-missing-field-initializers

392 # Unfortunately, this option can misfire very easily and unfixably.
393 CERRWARN +=     -_gcc=-Wno-array-bounds

395 # DEBUG v. -nd make for frequent unused variables, empty conditions, etc. in
396 # -nd builds
397 $(RELEASE_BUILD)CERRWARN += -_gcc=-Wno-unused
398 $(RELEASE_BUILD)CERRWARN += -_gcc=-Wno-empty-body

400 #
401 # turn warnings into errors (C++)
402 CCERRWARN=              -xwe

404 # C99 mode
405 C99_ENABLE=     -xc99=%all
406 C99_DISABLE=    -xc99=%none
407 C99MODE=        $(C99_DISABLE)
408 C99LMODE=       $(C99MODE:-xc99%=-Xc99%)

410 # In most places, assignments to these macros should be appended with +=
411 # (CPPFLAGS.master allows values to be prepended to CPPFLAGS).
412 sparc_CFLAGS=   $(sparc_XARCH) $(CCSTATICSYM)
413 sparcv9_CFLAGS= $(sparcv9_XARCH) -dalign $(CCVERBOSE) $(V9ABIWARN) $(CCREGSYM) \
414                 $(CCSTATICSYM)
415 i386_CFLAGS=    $(i386_XARCH)
416 amd64_CFLAGS=   $(amd64_XARCH)

418 sparc_ASFLAGS=  $(sparc_AS_XARCH)
419 sparcv9_ASFLAGS=$(sparcv9_AS_XARCH)
420 i386_ASFLAGS=   $(i386_AS_XARCH)
421 amd64_ASFLAGS=  $(amd64_AS_XARCH)

423 #
424 sparc_COPTFLAG=         -xO3
425 sparcv9_COPTFLAG=       -xO3
426 i386_COPTFLAG=          -O
427 amd64_COPTFLAG=         -xO3

429 COPTFLAG= $($(MACH)_COPTFLAG)
```



```
430 COPTFLAG64= $($(MACH64)_COPTFLAG)

432 # When -g is used, the compiler globalizes static objects
433 # (gives them a unique prefix). Disable that.
434 CNOGLOBAL= -W0,-noglobal

436 # Direct the Sun Studio compiler to use a static globalization prefix based on t
437 # name of the module rather than something unique. Otherwise, objects
438 # will not build deterministically, as subsequent compilations of identical
439 # source will yeild objects that always look different.
440 #
441 # In the same spirit, this will also remove the date from the N_OPT stab.
442 CGLOBALSTATIC= -W0,-xglobalstatic

444 # Sometimes we want all symbols and types in debugging information even
445 # if they aren't used.
446 CALLSYMS=       -W0,-xdbggen=no%usedonly

448 #
449 # Default debug format for Sun Studio 11 is dwarf, so force it to
450 # generate stabs.
451 #
452 DEBUGFORMAT=    -xdebugformat=stabs

454 #
455 # Flags used to build in debug mode for ctf generation.  Bugs in the Devpro
456 # compilers currently prevent us from building with cc-emitted DWARF.
457 #
458 CTF_FLAGS_sparc = -g -Wc,-Qiselect-T1 $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)
459 CTF_FLAGS_i386  = -g $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)

461 CTF_FLAGS_sparcv9       = $(CTF_FLAGS_sparc)
462 CTF_FLAGS_amd64         = $(CTF_FLAGS_i386)

464 # Sun Studio produces broken userland code when saving arguments.
465 $(__GNUC)CTF_FLAGS_amd64 += $(SAVEARGS)

467 CTF_FLAGS_32    = $(CTF_FLAGS_$(MACH)) $(DEBUGFORMAT)
468 CTF_FLAGS_64    = $(CTF_FLAGS_$(MACH64)) $(DEBUGFORMAT)
469 CTF_FLAGS       = $(CTF_FLAGS_32)

471 #
472 # Flags used with genoffsets
473 #
474 GOFLAGS = -_noecho \
475         $(CALLSYMS) \
476         $(CDWARFSTR)

478 OFFSETS_CREATE = $(GENOFFSETS) -s $(CTFSTABS) -r $(CTFCONVERT) \
479         $(CC) $(GOFLAGS) $(CFLAGS) $(CPPFLAGS)

481 OFFSETS_CREATE64 = $(GENOFFSETS) -s $(CTFSTABS) -r $(CTFCONVERT) \
482         $(CC) $(GOFLAGS) $(CFLAGS64) $(CPPFLAGS)

484 #
485 # tradeoff time for space (smaller is better)
486 #
487 sparc_SPACEFLAG         = -xspace -W0,-Lt
488 sparcv9_SPACEFLAG       = -xspace -W0,-Lt
489 i386_SPACEFLAG          = -xspace
490 amd64_SPACEFLAG         =

492 SPACEFLAG               = $($(MACH)_SPACEFLAG)
493 SPACEFLAG64             = $($(MACH64)_SPACEFLAG)

495 #
```

```
# The Sun Studio 11 compiler has changed the behaviour of integer
# wrap arounds and so a flag is needed to use the legacy behaviour
# (without this flag panics/hangs could be exposed within the source).
#
sparc_IROPTFLAG         = -W2,-xwrap_int
sparcv9_IROPTFLAG       = -W2,-xwrap_int
i386_IROPTFLAG          =
amd64_IROPTFLAG         =

IROPTFLAG               = $($(MACH)_IROPTFLAG)
IROPTFLAG64             = $($(MACH64)_IROPTFLAG)

sparc_XREGSFLAG         = -xregs=no%appl
sparcv9_XREGSFLAG       = -xregs=no%appl
i386_XREGSFLAG          =
amd64_XREGSFLAG         =

XREGSFLAG               = $($(MACH)_XREGSFLAG)
XREGSFLAG64             = $($(MACH64)_XREGSFLAG)

# dmake SOURCEDEBUG=yes ... enables source-level debugging information, and
# avoids stripping it.
SOURCEDEBUG     = $(POUND_SIGN)
SRCDBGBLD       = $(SOURCEDEBUG:yes=)

#
# These variables are intended ONLY for use by developers to safely pass extra
# flags to the compilers without unintentionally overriding Makefile-set
# flags.  They should NEVER be set to any value in a Makefile.
#
# They come last in the associated FLAGS variable such that they can
# explicitly override things if necessary, there are gaps in this, but it's
# the best we can manage.
#
CUSERFLAGS              =
CUSERFLAGS64            = $(CUSERFLAGS)
CCUSERFLAGS             =
CCUSERFLAGS64           = $(CCUSERFLAGS)

CSOURCEDEBUGFLAGS       =
CCSOURCEDEBUGFLAGS      =
$(SRCDBGBLD)CSOURCEDEBUGFLAGS   = -g -xs
$(SRCDBGBLD)CCSOURCEDEBUGFLAGS  = -g -xs

CFLAGS=         $(COPTFLAG) $($(MACH)_CFLAGS) $(SPACEFLAG) $(CCMODE) \
                $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG) \
                $(CGLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
                $(CUSERFLAGS)
CFLAGS64=       $(COPTFLAG64) $($(MACH64)_CFLAGS) $(SPACEFLAG64) $(CCMODE64) \
                $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG64) \
                $(CGLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
                $(CUSERFLAGS64)
#
# Flags that are used to build parts of the code that are subsequently
# run on the build machine (also known as the NATIVE_BUILD).
#
NATIVE_CFLAGS=  $(COPTFLAG) $($(NATIVE_MACH)_CFLAGS) $(CCMODE) \
                $(ILDOFF) $(CERRWARN) $(C99MODE) $($(NATIVE_MACH)_CCUNBOUND) \
                $(IROPTFLAG) $(CGLOBALSTATIC) $(CCNOAUTOINLINE) \
                $(CSOURCEDEBUGFLAGS) $(CUSERFLAGS)

DTEXTDOM=-DTEXT_DOMAIN=\"$(TEXT_DOMAIN)\"       # For messaging.
DTS_ERRNO=-D_TS_ERRNO
CPPFLAGS.master=$(DTEXTDOM) $(DTS_ERRNO) \
        $(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) $(ENVCPPFLAGS4) \
        $(ADJUNCT_PROTO:%=-I%/usr/include)
```



```
CPPFLAGS.native=$(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) \
                $(ENVCPPFLAGS4) -I$(NATIVE_ADJUNCT)/include
CPPFLAGS=       $(CPPFLAGS.master)
AS_CPPFLAGS=    $(CPPFLAGS.master)
JAVAFLAGS=      -deprecation

#
# For source message catalogue
#
.SUFFIXES: $(SUFFIXES) .i .po
MSGROOT= $(ROOT)/catalog
MSGDOMAIN= $(MSGROOT)/$(TEXT_DOMAIN)
MSGDOMAINPOFILE = $(MSGDOMAIN)/$(POFILE)
DCMSGDOMAIN= $(MSGROOT)/LC_TIME/$(TEXT_DOMAIN)
DCMSGDOMAINPOFILE = $(DCMSGDOMAIN)/$(DCFILE:.dc=.po)

CLOBBERFILES += $(POFILE) $(POFILES)
COMPILE.cpp= $(CC) -E -C $(CFLAGS) $(CPPFLAGS)
XGETTEXT= /usr/bin/xgettext
XGETFLAGS= -c TRANSLATION_NOTE
GNUXGETTEXT= /usr/gnu/bin/xgettext
GNUXGETFLAGS= --add-comments=TRANSLATION_NOTE --keyword=_ \
        --strict --no-location --omit-header
BUILD.po= $(XGETTEXT) $(XGETFLAGS) -d $(<F) $<.i ;\
        $(RM)   $@ ;\
        $(SED) "/^domain/d" < $(<F).po > $@ ;\
        $(RM) $(<F).po $<.i

#
# This is overwritten by local Makefile when PROG is a list.
#
POFILE= $(PROG).po

sparc_CCFLAGS=          -cg92 -compat=4 \
                        -Qoption ccfe -messages=no%anachronism \
                        $(CCERRWARN)
sparcv9_CCFLAGS=        $(sparcv9_XARCH) -dalign -compat=5 \
                        -Qoption ccfe -messages=no%anachronism \
                        -Qoption ccfe -features=no%conststrings \
                        $(CCCREGSYM) \
                        $(CCERRWARN)
i386_CCFLAGS=           -compat=4 \
                        -Qoption ccfe -messages=no%anachronism \
                        -Qoption ccfe -features=no%conststrings \
                        $(CCERRWARN)
amd64_CCFLAGS=          $(amd64_XARCH) -compat=5 \
                        -Qoption ccfe -messages=no%anachronism \
                        -Qoption ccfe -features=no%conststrings \
                        $(CCERRWARN)

sparc_CCOPTFLAG=        -O
sparcv9_CCOPTFLAG=      -O
i386_CCOPTFLAG=         -O
amd64_CCOPTFLAG=        -O

CCOPTFLAG=      $($(MACH)_CCOPTFLAG)
CCOPTFLAG64=    $($(MACH64)_CCOPTFLAG)
CCFLAGS=        $(CCOPTFLAG) $($(MACH)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
                $(CCUSERFLAGS)
CCFLAGS64=      $(CCOPTFLAG64) $($(MACH64)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
                $(CCUSERFLAGS64)

#
#
#
ELFWRAP_FLAGS   =
```

```
628 ELFWRAP_FLAGS64 =       -64

630 #
631 # Various mapfiles that are used throughout the build, and delivered to
632 # /usr/lib/ld.
633 #
634 MAPFILE.NED_i386 =      $(SRC)/common/mapfiles/common/map.noexdata
635 MAPFILE.NED_sparc =
636 MAPFILE.NED =           $(MAPFILE.NED_$(MACH))
637 MAPFILE.PGA =           $(SRC)/common/mapfiles/common/map.pagealign
638 MAPFILE.NES =           $(SRC)/common/mapfiles/common/map.noexstk
639 MAPFILE.FLT =           $(SRC)/common/mapfiles/common/map.filter
640 MAPFILE.LEX =           $(SRC)/common/mapfiles/common/map.lex.yy

642 #
643 # Generated mapfiles that are compiler specific, and used throughout the
644 # build.  These mapfiles are not delivered in /usr/lib/ld.
645 #
646 MAPFILE.NGB_sparc=      $(SRC)/common/mapfiles/gen/sparc_cc_map.noexeglobs
647 $(__GNUC64)MAPFILE.NGB_sparc= \
648                         $(SRC)/common/mapfiles/gen/sparc_gcc_map.noexeglobs
649 MAPFILE.NGB_sparcv9=    $(SRC)/common/mapfiles/gen/sparcv9_cc_map.noexeglobs
650 $(__GNUC64)MAPFILE.NGB_sparcv9= \
651                         $(SRC)/common/mapfiles/gen/sparcv9_gcc_map.noexeglobs
652 MAPFILE.NGB_i386=       $(SRC)/common/mapfiles/gen/i386_cc_map.noexeglobs
653 $(__GNUC64)MAPFILE.NGB_i386= \
654                         $(SRC)/common/mapfiles/gen/i386_gcc_map.noexeglobs
655 MAPFILE.NGB_amd64=      $(SRC)/common/mapfiles/gen/amd64_cc_map.noexeglobs
656 $(__GNUC64)MAPFILE.NGB_amd64= \
657                         $(SRC)/common/mapfiles/gen/amd64_gcc_map.noexeglobs
658 MAPFILE.NGB =           $(MAPFILE.NGB_$(MACH))

660 #
661 # A generic interface mapfile name, used by various dynamic objects to define
662 # the interfaces and interposers the object must export.
663 #
664 MAPFILE.INT =           mapfile-intf

666 #
667 # LDLIBS32 can be set in the environment to override the following assignment.
668 # LDLIBS64 can be set to override the assignment made in Makefile.master.64.
669 # These environment settings make sure that no libraries are searched outside
670 # of the local workspace proto area:
671 #       LDLIBS32=-YP,$ROOT/lib:$ROOT/usr/lib
672 #       LDLIBS64=-YP,$ROOT/lib/$MACH64:$ROOT/usr/lib/$MACH64
673 #
674 LDLIBS32 =      $(ENVLDLIBS1) $(ENVLDLIBS2) $(ENVLDLIBS3)
675 LDLIBS32 +=     $(ADJUNCT_PROTO:%=-L%/usr/lib -L%/lib)
676 LDLIBS.cmd =    $(LDLIBS32)
677 LDLIBS.lib =    $(LDLIBS32)
678 #
679 # Define compilation macros.
680 #
681 COMPILE.c=      $(CC) $(CFLAGS) $(CPPFLAGS) -c
682 COMPILE64.c=    $(CC) $(CFLAGS64) $(CPPFLAGS) -c
683 COMPILE.cc=     $(CCC) $(CCFLAGS) $(CPPFLAGS) -c
684 COMPILE64.cc=   $(CCC) $(CCFLAGS64) $(CPPFLAGS) -c
685 COMPILE.s=      $(AS) $(ASFLAGS) $(AS_CPPFLAGS)
686 COMPILE64.s=    $(AS) $(ASFLAGS) $($(MACH64)_AS_XARCH) $(AS_CPPFLAGS)
687 COMPILE.d=      $(DTRACE) -G -32
688 COMPILE64.d=    $(DTRACE) -G -64
689 COMPILE.b=      $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))
690 COMPILE64.b=    $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))

692 CLASSPATH=      .
693 COMPILE.java=   $(JAVAC) $(JAVAFLAGS) -classpath $(CLASSPATH)
```



```
695 #
696 # Link time macros
697 #
698 CCNEEDED                = -lC
699 CCEXTNEEDED             = -lCrun -lCstd
700 $(__GNUC)CCNEEDED       = -L$(GCCLIBDIR) -lstdc++ -lgcc_s
701 $(__GNUC)CCEXTNEEDED    = $(CCNEEDED)

703 LINK.c=         $(CC) $(CFLAGS) $(CPPFLAGS) $(LDFLAGS)
704 LINK64.c=       $(CC) $(CFLAGS64) $(CPPFLAGS) $(LDFLAGS)
705 NORUNPATH=      -norunpath -nolib
706 LINK.cc=        $(CCC) $(CCFLAGS) $(CPPFLAGS) $(NORUNPATH) \
707                 $(LDFLAGS) $(CCNEEDED)
708 LINK64.cc=      $(CCC) $(CCFLAGS64) $(CPPFLAGS) $(NORUNPATH) \
709                 $(LDFLAGS) $(CCNEEDED)

711 #
712 # lint macros
713 #
714 # Note that the undefine of __PRAGMA_REDEFINE_EXTNAME can be removed once
715 # ON is built with a version of lint that has the fix for 4484186.
716 #
717 ALWAYS_LINT_DEFS =      -errtags=yes -s
718 ALWAYS_LINT_DEFS +=     -erroff=E_PTRDIFF_OVERFLOW
719 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_NARROW_CONV
720 ALWAYS_LINT_DEFS +=     -U__PRAGMA_REDEFINE_EXTNAME
721 ALWAYS_LINT_DEFS +=     $(C99LMODE)
722 ALWAYS_LINT_DEFS +=     -errsecurity=$(SECLEVEL)
723 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_CREAT_WITHOUT_EXCL
724 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_FORBIDDEN_WARN_CREAT
725 # XX64 -- really only needed for amd64 lint
726 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_INT_TO_SMALL_INT
727 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_CONST_TO_SMALL_INT
728 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_TO_SMALL_INT
729 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_TO_PTR_FROM_INT
730 ALWAYS_LINT_DEFS +=     -erroff=E_COMP_INT_WITH_LARGE_INT
731 ALWAYS_LINT_DEFS +=     -erroff=E_INTEGRAL_CONST_EXP_EXPECTED
732 ALWAYS_LINT_DEFS +=     -erroff=E_PASS_INT_TO_SMALL_INT
733 ALWAYS_LINT_DEFS +=     -erroff=E_PTR_CONV_LOSES_BITS

735 # This forces lint to pick up note.h and sys/note.h from Devpro rather than
736 # from the proto area.  The note.h that ON delivers would disable NOTE().
737 ONLY_LINT_DEFS =        -I$(SPRO_VROOT)/prod/include/lint

739 SECLEVEL=       core
740 LINT.c=         $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS) $(CPPFLAGS) \
741                 $(ALWAYS_LINT_DEFS)
742 LINT64.c=       $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS64) $(CPPFLAGS) \
743                 $(ALWAYS_LINT_DEFS)
744 LINT.s=         $(LINT.c)

746 # For some future builds, NATIVE_MACH and MACH might be different.
747 # Therefore, NATIVE_MACH needs to be redefined in the
748 # environment as ‘uname -p‘ to override this macro.
749 #
750 # For now at least, we cross-compile amd64 on i386 machines.
751 NATIVE_MACH=    $(MACH:amd64=i386)

753 # Define native compilation macros
754 #

756 # Base directory where compilers are loaded.
757 # Defined here so it can be overridden by developer.
758 #
759 SPRO_ROOT=              $(BUILD_TOOLS)/SUNWspro
```

```
760 SPRO_VROOT=             $(SPRO_ROOT)/SS12
761 GNU_ROOT=               $(SFW_ROOT)

763 # Till SS12u1 formally becomes the NV CBE, LINT is hard
764 # coded to be picked up from the $SPRO_ROOT/sunstudio12.1/
765 # location. Impacted variables are sparc_LINT, sparcv9_LINT,
766 # i386_LINT, amd64_LINT.
767 # Reset them when SS12u1 is rolled out.
768 #

770 # Specify platform compiler versions for languages
771 # that we use (currently only c and c++).
772 #
773 sparc_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
774 $(__GNUC)sparc_CC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
775 sparc_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
776 $(__GNUC)sparc_CCC=     $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
777 sparc_CPP=              /usr/ccs/lib/cpp
778 sparc_AS=               /usr/ccs/bin/as -xregsym=no
779 sparc_LD=               /usr/ccs/bin/ld
780 sparc_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

782 sparcv9_CC=            $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
783 $(__GNUC64)sparcv9_CC=  $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
784 sparcv9_CCC=            $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
785 $(__GNUC64)sparcv9_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
786 sparcv9_CPP=            /usr/ccs/lib/cpp
787 sparcv9_AS=             /usr/ccs/bin/as -xregsym=no
788 sparcv9_LD=             /usr/ccs/bin/ld
789 sparcv9_LINT=           $(SPRO_ROOT)/sunstudio12.1/bin/lint

791 i386_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
792 $(__GNUC)i386_CC=       $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
793 i386_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
794 $(__GNUC)i386_CCC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
795 i386_CPP=              /usr/ccs/lib/cpp
796 i386_AS=               /usr/ccs/bin/as
797 $(__GNUC)i386_AS=       $(ONBLD_TOOLS)/bin/$(MACH)/aw
798 i386_LD=               /usr/ccs/bin/ld
799 i386_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

801 amd64_CC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
802 $(__GNUC64)amd64_CC=    $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
803 amd64_CCC=             $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
804 $(__GNUC64)amd64_CCC=   $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
805 amd64_CPP=             /usr/ccs/lib/cpp
806 amd64_AS=              $(ONBLD_TOOLS)/bin/$(MACH)/aw
807 amd64_LD=              /usr/ccs/bin/ld
808 amd64_LINT=            $(SPRO_ROOT)/sunstudio12.1/bin/lint

810 NATIVECC=              $($(NATIVE_MACH)_CC)
811 NATIVECCC=             $($(NATIVE_MACH)_CCC)
812 NATIVECPP=             $($(NATIVE_MACH)_CPP)
813 NATIVEAS=              $($(NATIVE_MACH)_AS)
814 NATIVELD=              $($(NATIVE_MACH)_LD)
815 NATIVELINT=            $($(NATIVE_MACH)_LINT)

817 #
818 # Makefile.master.64 overrides these settings
819 #
820 CC=                    $(NATIVECC)
821 CCC=                   $(NATIVECCC)
822 CPP=                   $(NATIVECPP)
823 AS=                    $(NATIVEAS)
824 LD=                    $(NATIVELD)
825 LINT=                  $(NATIVELINT)
```



```
827 # The real compilers used for this build
828 CW_CC_CMD=              $(CC) -_compiler
829 CW_CCC_CMD=             $(CCC) -_compiler
830 REAL_CC=                $(CW_CC_CMD:sh)
831 REAL_CCC=               $(CW_CCC_CMD:sh)

833 # Pass -Y flag to cpp (method of which is release-dependent)
834 CCYFLAG=                -Y I,

836 BDIRECT=        -Bdirect
837 BDYNAMIC=       -Bdynamic
838 BLOCAL=         -Blocal
839 BNODIRECT=      -Bnodirect
840 BREDUCE=        -Breduce
841 BSTATIC=        -Bstatic

843 ZDEFS=          -zdefs
844 ZDIRECT=        -zdirect
845 ZIGNORE=        -zignore
846 ZINITFIRST=     -zinitfirst
847 ZINTERPOSE=     -zinterpose
848 ZLAZYLOAD=      -zlazyload
849 ZLOADFLTR=      -zloadfltr
850 ZMULDEFS=       -zmuldefs
851 ZNODEFAULTLIB=  -znodefaultlib
852 ZNODEFS=        -znodefs
853 ZNODELETE=      -znodelete
854 ZNODLOPEN=      -znodlopen
855 ZNODUMP=        -znodump
856 ZNOLAZYLOAD=    -znolazyload
857 ZNOLDYNSYM=     -znoldynsym
858 ZNORELOC=       -znoreloc
859 ZNOVERSION=     -znoversion
860 ZRECORD=        -zrecord
861 ZREDLOCSYM=     -zredlocsym
862 ZTEXT=          -ztext
863 ZVERBOSE=       -zverbose

865 GSHARED=        -G
866 CCMT=           -mt

868 # Handle different PIC models on different ISAs
869 # (May be overridden by lower-level Makefiles)

871 sparc_C_PICFLAGS =      -K pic
872 sparcv9_C_PICFLAGS =    -K pic
873 i386_C_PICFLAGS =       -K pic
874 amd64_C_PICFLAGS =      -K pic
875 C_PICFLAGS =            $($(MACH)_C_PICFLAGS)
876 C_PICFLAGS64 =          $($(MACH64)_C_PICFLAGS)

878 sparc_C_BIGPICFLAGS =   -K PIC
879 sparcv9_C_BIGPICFLAGS = -K PIC
880 i386_C_BIGPICFLAGS =    -K PIC
881 amd64_C_BIGPICFLAGS =   -K PIC
882 C_BIGPICFLAGS =         $($(MACH)_C_BIGPICFLAGS)
883 C_BIGPICFLAGS64 =       $($(MACH64)_C_BIGPICFLAGS)

885 # CC requires there to be no space between '-K' and 'pic' or 'PIC'.
886 sparc_CC_PICFLAGS =     -Kpic
887 sparcv9_CC_PICFLAGS =   -KPIC
888 i386_CC_PICFLAGS =      -Kpic
889 amd64_CC_PICFLAGS =     -Kpic
890 CC_PICFLAGS =           $($(MACH)_CC_PICFLAGS)
891 CC_PICFLAGS64 =         $($(MACH64)_CC_PICFLAGS)
```

```
AS_PICFLAGS=            $(C_PICFLAGS)
AS_BIGPICFLAGS=         $(C_BIGPICFLAGS)

#
# Default label for CTF sections
#
CTFCVTFLAGS=            -i -L VERSION
$(SRCDBGBLD)CTFCVTFLAGS         += -g

#
# Override to pass module-specific flags to ctfmerge.  Currently used only by
# krtld to turn on fuzzy matching, and source-level debugging to inhibit
# stripping.
#
CTFMRGFLAGS=
$(SRCDBGBLD)CTFMRGFLAGS         += -g


CTFCONVERT_O            = $(CTFCONVERT) $(CTFCVTFLAGS) $@

ELFSIGN_O=      $(TRUE)
ELFSIGN_CRYPTO= $(ELFSIGN_O)
ELFSIGN_OBJECT= $(ELFSIGN_O)

# Rules (normally from make.rules) and macros which are used for post
# processing files. Normally, these do stripping of the comment section
# automatically.
#    RELEASE_CM:        Should be editted to reflect the release.
#    POST_PROCESS_O:    Post-processing for '.o' files.
#    POST_PROCESS_A:    Post-processing for '.a' files (currently null).
#    POST_PROCESS_SO:   Post-processing for '.so' files.
#    POST_PROCESS:      Post-processing for executable files (no suffix).
# Note that these macros are not completely generalized as they are to be
# used with the file name to be processed following.
#
# It is left as an exercise to Release Engineering to embellish the generation
# of the release comment string.
#
#       If this is a standard development build:
#               compress the comment section (mcs -c)
#               add the standard comment (mcs -a $(RELEASE_CM))
#               add the development specific comment (mcs -a $(DEV_CM))
#
#       If this is an installation build:
#               delete the comment section (mcs -d)
#               add the standard comment (mcs -a $(RELEASE_CM))
#               add the development specific comment (mcs -a $(DEV_CM))
#
#       If this is an release build:
#               delete the comment section (mcs -d)
#               add the standard comment (mcs -a $(RELEASE_CM))
#
# The following list of macros are used in the definition of RELEASE_CM
# which is used to label all binaries in the build:
#
#       RELEASE         Specific release of the build, eg: 5.2
#       RELEASE_MAJOR   Major version number part of $(RELEASE)
#       RELEASE_MINOR   Minor version number part of $(RELEASE)
#       VERSION         Version of the build (alpha, beta, Generic)
#       PATCHID         If this is a patch this value should contain
#                       the patchid value (eg: "Generic 100832-01"), otherwise
#                       it will be set to $(VERSION)
#       RELEASE_DATE    Date of the Release Build
#       PATCH_DATE      Date the patch was created, if this is blank it
#                       will default to the RELEASE_DATE
```



```
#
RELEASE_MAJOR=  5
RELEASE_MINOR=  11
RELEASE=        $(RELEASE_MAJOR).$(RELEASE_MINOR)
VERSION=        SunOS Development
PATCHID=        $(VERSION)
RELEASE_DATE=   release date not set
PATCH_DATE=     $(RELEASE_DATE)
RELEASE_CM=     "@($(POUND_SIGN))SunOS $(RELEASE) $(PATCHID) $(PATCH_DATE)"
DEV_CM=         "@($(POUND_SIGN))SunOS Internal Development: non-nightly build"

PROCESS_COMMENT=                    @?${MCS} -c -a $(RELEASE_CM) -a $(DEV_CM)
$(STRIP_COMMENTS)PROCESS_COMMENT=   @?${MCS} -d -a $(RELEASE_CM) -a $(DEV_CM)
$(RELEASE_BUILD)PROCESS_COMMENT=    @?${MCS} -d -a $(RELEASE_CM)

STRIP_STABS=                        :
$(RELEASE_BUILD)STRIP_STABS=        $(STRIP) -x $@
$(SRCDBGBLD)STRIP_STABS=            :

POST_PROCESS_O=         $(PROCESS_COMMENT) $@
POST_PROCESS_A=
POST_PROCESS_SO=        $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
                        $(ELFSIGN_OBJECT)
POST_PROCESS=           $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
                        $(ELFSIGN_OBJECT)

#
# chk4ubin is a tool that inspects a module for a symbol table
# ELF section size which can trigger an OBP bug on older platforms.
# This problem affects only specific sun4u bootable modules.
#
CHK4UBIN=       $(ONBLD_TOOLS)/bin/$(MACH)/chk4ubin
CHK4UBINFLAGS=
CHK4UBINARY=    $(CHK4UBIN) $(CHK4UBINFLAGS) $@

#
# PKGARCHIVE specifies the default location where packages should be
# placed if built.
#
$(RELEASE_BUILD)PKGARCHIVESUFFIX=       -nd
PKGARCHIVE=$(SRC)/../../packages/$(MACH)/nightly$(PKGARCHIVESUFFIX)

#
# The repositories will be created with these publisher settings.  To
# update an image to the resulting repositories, this must match the
# publisher name provided to "pkg set-publisher."
#
PKGPUBLISHER_REDIST=    on-nightly
PKGPUBLISHER_NONREDIST= on-extra

#       Default build rules which perform comment section post-processing.
#
.c:
        $(LINK.c) -o $@ $< $(LDLIBS)
        $(POST_PROCESS)
.c.o:
        $(COMPILE.c) $(OUTPUT_OPTION) $< $(CTFCONVERT_HOOK)
        $(POST_PROCESS_O)
.c.a:
        $(COMPILE.c) -o $% $<
        $(PROCESS_COMMENT) $%
        $(AR) $(ARFLAGS) $@ $%
        $(RM) $%
.s.o:
        $(COMPILE.s) -o $@ $<
        $(POST_PROCESS_O)
```

```
.s.a:
        $(COMPILE.s) -o $% $<
        $(PROCESS_COMMENT) $%
        $(AR) $(ARFLAGS) $@ $%
        $(RM) $%
.cc:
        $(LINK.cc) -o $@ $< $(LDLIBS)
        $(POST_PROCESS)
.cc.o:
        $(COMPILE.cc) $(OUTPUT_OPTION) $<
        $(POST_PROCESS_O)
.cc.a:
        $(COMPILE.cc) -o $% $<
        $(AR) $(ARFLAGS) $@ $%
        $(PROCESS_COMMENT) $%
        $(RM) $%
.y:
        $(YACC.y) $<
        $(LINK.c) -o $@ y.tab.c $(LDLIBS)
        $(POST_PROCESS)
        $(RM) y.tab.c
.y.o:
        $(YACC.y) $<
        $(COMPILE.c) -o $@ y.tab.c $(CTFCONVERT_HOOK)
        $(POST_PROCESS_O)
        $(RM) y.tab.c
.l:
        $(RM) $*.c
        $(LEX.l) $< > $*.c
        $(LINK.c) -o $@ $*.c -ll $(LDLIBS)
        $(POST_PROCESS)
        $(RM) $*.c
.l.o:
        $(RM) $*.c
        $(LEX.l) $< > $*.c
        $(COMPILE.c) -o $@ $*.c $(CTFCONVERT_HOOK)
        $(POST_PROCESS_O)
        $(RM) $*.c

.bin.o:
        $(COMPILE.b) -o $@ $<
        $(POST_PROCESS_O)

.java.class:
        $(COMPILE.java) $<

# Bourne and Korn shell script message catalog build rules.
# We extract all gettext strings with sed(1) (being careful to permit
# multiple gettext strings on the same line), weed out the dups, and
# build the catalogue with awk(1).

.sh.po .ksh.po:
        $(SED) -n -e ":a"                                       \
                  -e "h"                                                \
                  -e "s/.*gettext *\(\"[^\"]*\"\).*/\1/p"       \
                  -e "x"                                                \
                  -e "s/\(.*\)gettext *\"[^\"]*\"\(.*\)/\1\2/"  \
                  -e "t a"                                              \
                $< | sort -u | awk '{ print "msgid\t" $$0 "\nmsgstr" }' > $@

#
# Python and Perl executable and message catalog build rules.
#
.SUFFIXES: .pl .pm .py .pyc

.pl:
```



```
        $(RM) $@;
        $(SED) -e "s@TEXT_DOMAIN@\"$(TEXT_DOMAIN)\"@" $< > $@;
        $(CHMOD) +x $@

.py:
        $(RM) $@; $(CAT) $< > $@; $(CHMOD) +x $@

.py.pyc:
        $(RM) $@
        $(PYTHON) -mpy_compile $<
        @[ $(<)c = $@ ] || $(MV) $(<)c $@

.py.po:
        $(GNUXGETTEXT) $(GNUXGETFLAGS) -d $(<F:%.py=%) $< ;

.pl.po .pm.po:
        $(XGETTEXT) $(XGETFLAGS) -d $(<F) $< ;
        $(RM)   $@ ;
        $(SED) "/^domain/d" < $(<F).po > $@ ;
        $(RM) $(<F).po

#
# When using xgettext, we want messages to go to the default domain,
# rather than the specified one.  This special version of the
# COMPILE.cpp macro effectively prevents expansion of TEXT_DOMAIN,
# causing xgettext to put all messages into the default domain.
#
CPPFORPO=$(COMPILE.cpp:\"$(TEXT_DOMAIN)\"=TEXT_DOMAIN)

.c.i:
        $(CPPFORPO) $< > $@

.h.i:
        $(CPPFORPO) $< > $@

.y.i:
        $(YACC) -d $<
        $(CPPFORPO) y.tab.c  > $@
        $(RM) y.tab.c

.l.i:
        $(LEX) $<
        $(CPPFORPO) lex.yy.c  > $@
        $(RM) lex.yy.c

.c.po:
        $(CPPFORPO) $< > $<.i
        $(BUILD.po)

.y.po:
        $(YACC) -d $<
        $(CPPFORPO) y.tab.c  > $<.i
        $(BUILD.po)
        $(RM) y.tab.c

.l.po:
        $(LEX) $<
        $(CPPFORPO) lex.yy.c  > $<.i
        $(BUILD.po)
        $(RM) lex.yy.c

#
# Rules to perform stylistic checks
#
.SUFFIXES: .x .xml .check .xmlchk
```

```
1156 .h.check:
1157         $(DOT_H_CHECK)

1159 .x.check:
1160         $(DOT_X_CHECK)

1162 .xml.xmlchk:
1163         $(MANIFEST_CHECK)

1165 #
1166 # Include rules to render automated sccs get rules "safe".
1167 #
1168 include $(SRC)/Makefile.noget
```

```
**********************************************************
   10856 Wed Oct 16 17:41:04 2013
new/usr/src/cmd/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

# Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
# Copyright 2011 Joyent, Inc.  All rights reserved.
# Copyright (c) 2012 by Delphix. All rights reserved.
# Copyright (c) 2013 DEY Storage Systems, Inc. All rights reserved.

include ../Makefile.master

#
#       Note that the commands 'agents', 'lp', 'perl', and 'man' are first in
#       the list, violating alphabetical order.  This is because they are very
#       long-running and should be given the most wall-clock time for a
#       parallel build.
#
#       Commands in the FIRST_SUBDIRS list are built before starting the build
#       of other commands.  Currently this includes only 'isaexec' and
#       'platexec'.  This is necessary because $(ROOT)/usr/lib/isaexec or
#       $(ROOT)/usr/lib/platexec must exist when some other commands are built
#       because their 'make install' creates a hard link to one of them.
#
#       Commands are listed one per line so that TeamWare can auto-merge most
#       changes.
#

FIRST_SUBDIRS=          \
        isaexec         \
        platexec

COMMON_SUBDIRS=         \
        allocate        \
        availdevs       \
        lp              \
        perl            \
        man             \
        Adm             \
        abi             \
        adbgen          \
```



```
        acct            \
        acctadm         \
        arch            \
        asa             \
        ast             \
        audio           \
        auths           \
        autopush        \
        avs             \
        awk             \
        awk_xpg4        \
        backup          \
        banner          \
        bart            \
        basename        \
        bc              \
        bdiff           \
        beadm           \
        bfs             \
        bnu             \
        boot            \
        busstat         \
        cal             \
        calendar        \
        captoinfo       \
        cat             \
        cdrw            \
        cfgadm          \
        checkeq         \
        checknr         \
        chgrp           \
        chmod           \
        chown           \
        chroot          \
        clear           \
        clinfo          \
        cmd-crypto      \
        cmd-inet        \
        col             \
        compress        \
        consadm         \
        coreadm         \
        cpio            \
        cpc             \
        cron            \
        crypt           \
        csh             \
        csplit          \
        ctrun           \
        ctstat          \
        ctwatch         \
        datadm          \
        date            \
        dc              \
        dd              \
        deroff          \
        devfsadm        \
        syseventd       \
        devctl          \
        devinfo         \
        devmgmt         \
        devprop         \
        dfs.cmds        \
        diff            \
        diff3           \
        diffmk          \
```

```
125         dircmp          \
126         dirname         \
127         dis             \
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         fmtmsg          \
168         fold            \
169         format          \
170         fs.d            \
171         fstyp           \
172         fuser           \
173         fwflash         \
174         gcore           \
175         gencat          \
176         geniconvtbl     \
177         genmsg          \
178         getconf         \
179         getdevpolicy    \
180         getent          \
181         getfacl         \
182         getmajor        \
183         getopt          \
184         gettext         \
185         gettxt          \
186         grep            \
187         grep_xpg4       \
188         groups          \
189         grpck           \
190         gss             \
```



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

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



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

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



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

472 $(CLOSED_BUILD)COMMON_SUBDIRS += \
473         $(CLOSED)/cmd/iconv     \
474         $(CLOSED)/cmd/ksh       \
475         $(CLOSED)/cmd/localedef \
476         $(CLOSED)/cmd/more_xpg4 \
477         $(CLOSED)/cmd/mtst      \
478         $(CLOSED)/cmd/od        \
479         $(CLOSED)/cmd/patch     \
480         $(CLOSED)/cmd/pax       \
481         $(CLOSED)/cmd/printf    \
482         $(CLOSED)/cmd/sed       \
483         $(CLOSED)/cmd/sed_xpg4

472 i386_SUBDIRS=          \
473         acpihpd         \
474         addbadsec       \
475         biosdev         \
476         diskscan        \
477         lms             \
478         ntfsprogs       \
479         parted          \
480         rtc             \
481         ucodeadm        \
482         xvm

484 sparc_SUBDIRS=          \
485         cvcd            \
486         dcs             \
487         device_remap    \
488         drd             \
489         fruadm          \
490         ldmad           \
491         oplhpd          \
492         prtdscp         \
493         prtfru          \
494         scadm           \
495         sckmd           \
496         sf880drd        \
497         virtinfo        \
498         vntsd

500 #
501 # Commands that are messaged.  Note that 'lp' and 'man' come first
502 # (see previous comment about 'lp' and 'man').
503 #
504 MSGSUBDIRS=             \
505         lp              \
506         man             \
507         abi             \
```

```
508         acctadm         \
509         allocate        \
510         asa             \
511         audio           \
512         audit           \
513         auditconfig     \
514         auditd          \
515         auditrecord     \
516         auditset        \
517         auths           \
518         autopush        \
519         avs             \
520         awk             \
521         awk_xpg4        \
522         backup          \
523         banner          \
524         bart            \
525         basename        \
526         beadm           \
527         bnu             \
528         busstat         \
529         cal             \
530         cat             \
531         cdrw            \
532         cfgadm          \
533         checkeq         \
534         checknr         \
535         chgrp           \
536         chmod           \
537         chown           \
538         cmd-crypto      \
539         cmd-inet        \
540         col             \
541         compress        \
542         consadm         \
543         coreadm         \
544         cpio            \
545         cpc             \
546         cron            \
547         csh             \
548         csplit          \
549         ctrun           \
550         ctstat          \
551         ctwatch         \
552         datadm          \
553         date            \
554         dc              \
555         dcs             \
556         dd              \
557         deroff          \
558         devfsadm        \
559         dfs.cmds        \
560         diff            \
561         diffmk          \
562         dladm           \
563         dlstat          \
564         du              \
565         dumpcs          \
566         ed              \
567         eject           \
568         env             \
569         eqn             \
570         expand          \
571         expr            \
572         fcinfo          \
573         fgrep           \
```



```
574         file            \
575         filesync        \
576         find            \
577         flowadm         \
578         flowstat        \
579         fm              \
580         fold            \
581         fs.d            \
582         fwflash         \
583         geniconvtbl     \
584         genmsg          \
585         getconf         \
586         getent          \
587         gettext         \
588         gettxt          \
589         grep            \
590         grep_xpg4       \
591         grpck           \
592         gss             \
593         halt            \
594         head            \
595         hostname        \
596         hotplug         \
597         id              \
598         idmap           \
599         isaexec         \
600         iscsiadm        \
601         iscsid          \
602         isns            \
603         itadm           \
604         kbd             \
605         krb5            \
606         ksh             \
607         last            \
608         ldap            \
609         ldapcachemgr    \
610         lgrpinfo        \
611         locale          \
612         lofiadm         \
613         logadm          \
614         logger          \
615         logins          \
616         ls              \
617         luxadm          \
618         lvm             \
619         mailx           \
620         mesg            \
621         mkdir           \
622         mkpwdict        \
623         mktemp          \
624         more            \
625         mpathadm        \
626         msgfmt          \
627         mv              \
628         ndmpadm         \
629         ndmpstat        \
630         newgrp          \
631         newtask         \
632         nice            \
633         nohup           \
634         oawk            \
635         pack            \
636         passwd          \
637         passmgmt        \
638         pathchk         \
639         pfexec          \
```

```
640         pg              \
641         pgrep           \
642         picl            \
643         pools           \
644         power           \
645         pr              \
646         praudit         \
647         print           \
648         profiles        \
649         projadd         \
650         projects        \
651         prstat          \
652         prtdiag         \
653         ps              \
654         psrinfo         \
655         ptools          \
656         pwconv          \
657         pwd             \
658         pyzfs           \
659         raidctl         \
660         ramdiskadm      \
661         rcap            \
662         rcm_daemon      \
663         refer           \
664         regcmp          \
665         renice          \
666         roles           \
667         rm              \
668         rmdir           \
669         rmformat        \
670         rmmount         \
671         rmvolmgr        \
672         sasinfo         \
673         sbdadm          \
674         scadm           \
675         script          \
676         scsi            \
677         sdiff           \
678         sdpadm          \
679         sgs             \
680         sh              \
681         shcomp          \
682         smbsrv          \
683         sort            \
684         split           \
685         srptadm         \
686         ssh             \
687         stat            \
688         stmfadm         \
689         stmsboot        \
690         strings         \
691         su              \
692         svc             \
693         svr4pkg         \
694         swap            \
695         syseventadm     \
696         syseventd       \
697         tabs            \
698         tar             \
699         tbl             \
700         time            \
701         tnf             \
702         touch           \
703         tput            \
704         troff           \
705         tsol            \
```



```
706         tty             \
707         ttymon          \
708         tzreload        \
709         ul              \
710         uname           \
711         units           \
712         unlink          \
713         unpack          \
714         userattr        \
715         valtools        \
716         vgrind          \
717         vi              \
718         volcheck        \
719         volrmmount      \
720         vrrpadm         \
721         vscan           \
722         w               \
723         who             \
724         whodo           \
725         wracct          \
726         write           \
727         wusbadm         \
728         xargs           \
729         yppasswd        \
730         zdump           \
731         zfs             \
732         zic             \
733         zlogin          \
734         zoneadm         \
735         zoneadmd        \
736         zonecfg         \
737         zonename        \
738         zpool           \
739         zonestat

754 $(CLOSED_BUILD)MSGSUBDIRS += \
755         $(CLOSED)/cmd/iconv     \
756         $(CLOSED)/cmd/ksh       \
757         $(CLOSED)/cmd/localedef \
758         $(CLOSED)/cmd/more_xpg4 \
759         $(CLOSED)/cmd/od        \
760         $(CLOSED)/cmd/patch     \
761         $(CLOSED)/cmd/pax       \
762         $(CLOSED)/cmd/printf    \
763         $(CLOSED)/cmd/sed       \
764         $(CLOSED)/cmd/sed_xpg4

741 sparc_MSGSUBDIRS=      \
742         fruadm          \
743         prtdscp         \
744         prtfru          \
745         virtinfo        \
746         vntsd

748 i386_MSGSUBDIRS=       \
749         ucodeadm

751 #
752 # commands that use dcgettext for localized time, LC_TIME
753 #
754 DCSUBDIRS=              \
755         cal             \
756         cfgadm          \
757         diff            \
758         ls              \
759         pr              \
```

```
760         ps              \
761         tar             \
762         w               \
763         who             \
764         whodo           \
765         write

792 $(CLOSED_BUILD)DCSUBDIRS += \
793         $(CLOSED)/cmd/pax

767 #
768 # commands that belong only to audit.
769 #
770 AUDITSUBDIRS=           \
771         amt             \
772         audit           \
773         audit_warn      \
774         auditconfig     \
775         auditd          \
776         auditrecord     \
777         auditreduce     \
778         auditset        \
779         auditstat       \
780         praudit

782 #
783 # commands not owned by the systems group
784 #
785 BWOSDIRS=


788 all :=          TARGET = all
789 install :=      TARGET = install
790 clean :=        TARGET = clean
791 clobber :=      TARGET = clobber
792 lint :=         TARGET = lint
793 _msg :=         TARGET = _msg
794 _dc :=          TARGET = _dc

796 .KEEP_STATE:

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

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

802 all install clean clobber lint: $(FIRST_SUBDIRS) .WAIT $(SUBDIRS) \
803         $(AUDITSUBDIRS)

805 #
806 # Manifests cannot be checked in parallel, because we are using
807 # the global repository that is in $(SRC)/cmd/svc/seed/global.db.
808 # For this reason, to avoid .PARALLEL and .NO_PARALLEL conflicts,
809 # we spawn off a sub-make to perform the non-parallel 'make check'
810 #
811 check:
812         $(MAKE) -f Makefile.check check

814 #
815 # The .WAIT directive works around an apparent bug in parallel make.
816 # Evidently make was getting the target _msg vs. _dc confused under
817 # some level of parallelization, causing some of the _dc objects
818 # not to be built.
819 #
820 _msg: $(MSGSUBDIRS) $($(MACH)_MSGSUBDIRS) .WAIT _dc

822 _dc: $(DCSUBDIRS)
```



```
824 #
825 # Dependencies
826 #
827 fs.d: fstyp
828 ksh:    shcomp isaexec
829 mdb:    terminfo
830 print:  lp

832 $(FIRST_SUBDIRS) $(BWOSDIRS) $(SUBDIRS) $(AUDITSUBDIRS): FRC
833         @if [ -f $@/Makefile  ]; then \
834                 cd $@; pwd; $(MAKE) $(TARGET); \
835         else \
836                 true; \
837         fi

839 FRC:
```

```
**********************************************************
    1429 Wed Oct 16 17:41:04 2013
new/usr/src/cmd/cmd-crypto/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright (c) 2003, 2010, Oracle and/or its affiliates. All rights reserved.
#

include ../Makefile.cmd

SUBDIRS1 = \
        cryptoadm \
        decrypt \
        digest \
        elfsign \
        kmfcfg \
        pktool \
        tpmadm

$(CLOSED_BUILD)SUBDIRS1 += \
        $(CLOSED)/cmd/cmd-crypto/kcfd

SUBDIRS2 = \
        etc \
        scripts

$(CLOSED_BUILD)SUBDIRS2 += \
        $(CLOSED)/cmd/cmd-crypto/etc

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber
lint:=          TARGET= lint
_msg:=          TARGET= _msg

.KEEP_STATE:

all clean clobber lint _msg: $(SUBDIRS1) $($(MACH)_SUBDIRS)

install: $(SUBDIRS1) $(SUBDIRS2) $($(MACH)_SUBDIRS)

$(SUBDIRS1) $(SUBDIRS2) $($(MACH)_SUBDIRS) : FRC
```



```
        @cd $@; pwd; $(MAKE) $(MFLAGS) $(TARGET)

FRC:
```

```
**********************************************************
    1929 Wed Oct 16 17:41:04 2013
new/usr/src/cmd/cmd-inet/usr.lib/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1996, 2010, Oracle and/or its affiliates. All rights reserved.
#

SUBDIRS=        bridged dhcp dsvclockd ilbd in.chargend in.daytimed \
                in.discardd in.echod in.dhcpd in.mpathd in.ndpd \
                in.ripngd in.timed inetd mdnsd ncaconfd pppoe \
                slpd vrrpd wanboot wpad

MSGSUBDIRS=     dsvclockd ilbd in.dhcpd inetd ncaconfd vrrpd wanboot

include ../../Makefile.cmd
include ./Makefile.lib

$(CLOSED_BUILD)SUBDIRS += \
        $(CLOSED)/cmd/cmd-inet/usr.lib/ike-certutils \
        $(CLOSED)/cmd/cmd-inet/usr.lib/in.iked

POFILES=        dsvclockd/dsvclockd.po in.dhcpd/in.dhcpd.po \
                inetd/inetd.po ncaconfd/ncaconfd.po vrrpd/vrrpd.po \
                wanboot/wanboot.po
POFILE=         usr.lib.po

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber
lint:=          TARGET= lint
_msg:=          TARGET= _msg

.KEEP_STATE:

all clean clobber lint: $(SUBDIRS)

install: $(SUBDIRS)
        -$(RM) $(ROOTLIBINET)/in.iked
        -$(LN) $(ISAEXEC) $(ROOTLIBINET)/in.iked
```



```
_msg: $(MSGSUBDIRS)

#
# The reason this rule checks for the existence of the
# Makefile is that some of the directories do not exist
# in our exportable source builds or in OpenSolaris.
#
$(SUBDIRS): FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(TARGET); \
        else \
                true; \
        fi

FRC:
```

```
**********************************************************
    1352 Wed Oct 16 17:41:04 2013
new/usr/src/cmd/fwflash/plugins/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 2009 Sun Microsystems, Inc.  All rights reserved.
  23 # Use is subject to license terms.
  24 #
  25 # cmd/fwflash/plugins
  26 #
  27 include $(SRC)/Makefile.master

  29 COMMON_SUBDIRS= $(MACH)
  30 $(CLOSED_BUILD)COMMON_SUBDIRS += $(CLOSED)/cmd/fwflash/plugins

  31 SUBDIRS=        $(COMMON_SUBDIRS)

  33 all :=          TARGET= all
  34 install :=      TARGET= install
  35 clean :=        TARGET= clean
  36 clobber :=      TARGET= clobber
  37 lint :=         TARGET= lint
  38 _msg :=         TARGET= _msg
  39 msg :=          TARGET= msg


  42 .KEEP_STATE:

  44 all clean clobber install lint msg _msg:               $(SUBDIRS)


  47 $(SUBDIRS): FRC
  48         @if [ -f $@/Makefile ]; then \
  49                 cd $@; pwd; $(MAKE) $(TARGET); \
  50         else \
  51                 true; \
  52         fi

  54 FRC:
```

```
**********************************************************
    1175 Wed Oct 16 17:41:05 2013
new/usr/src/cmd/mdb/intel/amd64/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

include ../../Makefile.common

MODULES = $(COMMON_MODULES_PROC) $(COMMON_MODULES_KVM) uhci

$(CLOSED_BUILD)MODULES += \
        $(CLOSED_COMMON_MODULES_KVM:%=$(CLOSED)/cmd/mdb/intel/amd64/%)

SUBDIRS = mdb mdb_ks kmdb libstandctf libstand .WAIT $(MODULES)

include ../../Makefile.subdirs

.PARALLEL: $(SUBDIRS)

# inter-module dependencies
kmdb: libstand libstandctf mdb_ks
```

****
1210 Wed Oct 16 17:41:05 2013
new/usr/src/cmd/mdb/intel/ia32/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
****

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

include ../../Makefile.common

MODULES = $(COMMON_MODULES_PROC) $(COMMON_MODULES_PROC_32BIT) \
    $(COMMON_MODULES_KVM) uhci

$(CLOSED_BUILD)MODULES += \
        $(CLOSED_COMMON_MODULES_KVM:%=$(CLOSED)/cmd/mdb/intel/ia32/%)

SUBDIRS = mdb mdb_ks kmdb libstandctf libstand .WAIT $(MODULES)

include ../../Makefile.subdirs

.PARALLEL: $(SUBDIRS)

# inter-module dependencies
kmdb: libstand libstandctf mdb_ks
```

```
**********************************************************
    1237 Wed Oct 16 17:41:05 2013
new/usr/src/cmd/mdb/sparc/v9/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"

include ../../Makefile.common

MODULES = \
        $(COMMON_MODULES_PROC) \
        $(COMMON_MODULES_KVM) \
        intr \
        ssd

$(CLOSED_BUILD)MODULES += \
        $(CLOSED_COMMON_MODULES_KVM:%=$(CLOSED)/cmd/mdb/sparc/v9/%)
$(CLOSED_BUILD)MODULES += \
        $(CLOSED)/cmd/mdb/sparc/v9/isp

#
# a "$(MODULES): kmdb" rule would seem to do the trick but, for some reason,
# it serializes $(MODULES).
#
SUBDIRS = mdb mdb_ks kmdb libstandctf libstand .WAIT $(MODULES)

include ../../Makefile.subdirs

.PARALLEL: $(SUBDIRS)
```

```
**********************************************************
    1735 Wed Oct 16 17:41:05 2013
new/usr/src/cmd/tsol/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"
#
# cmd/tsol/Makefile
#

include ../Makefile.cmd

SUBDIRS =               \
        atohexlabel     \
        getlabel        \
        demo            \
        misc            \
        zones           \
        labeld          \
        updatehome      \
        plabel          \
        setlabel        \
        tnchkdb         \
        tninfo          \
        tnctl           \
        tnd             \
        tsol-zones      \
        getzonepath     \
        hextoalabel     \
        lslabels


MSGSUBDIRS =            \
        getlabel        \
        setlabel        \
        tnchkdb         \
        tninfo          \
        tnctl           \
        tnd             \
```



```
        getzonepath

$(CLOSED_BUILD)SUBDIRS += $(CLOSED)/cmd/tsol/chk_encodings \
        $(CLOSED)/cmd/tsol/labeld

$(CLOSED_BUILD)CLOSED_MSGSUBDIRS = $(CLOSED)/cmd/tsol/chk_encodings \
        $(CLOSED)/cmd/tsol/labeld

#
# for messaging catalog
#
POFILE= tsol-cmd.po
POFILES= $(MSGSUBDIRS:%=%/%.po)

all     :=      TARGET = all
install :=      TARGET = install
clean   :=      TARGET = clean
clobber :=      TARGET = clobber
lint    :=      TARGET = lint
_msg    :=      TARGET = _msg

.KEEP_STATE:

.PARALLEL:      $(SUBDIRS)

all install clean clobber lint: $(SUBDIRS)

$(POFILE): $(MSGSUBDIRS) $(CLOSED_MSGSUBDIRS)
        $(RM) $(POFILE)
        $(CAT) $(POFILES) > $(POFILE)

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

FRC:

include Makefile.targ
```

```
**********************************************************
   13431 Wed Oct 16 17:41:05 2013
new/usr/src/lib/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

# Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
# Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2012 by Delphix. All rights reserved.
# Copyright (c) 2012, Joyent, Inc. All rights reserved.
# Copyright (c) 2013 Gary Mills

include ../Makefile.master

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

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



```
SUBDIRS += \
        libc                    .WAIT   \
        ../cmd/sgs/libelf       .WAIT   \
        c_synonyms                      \
        libmd                           \
        libmd5                          \
        librsm                          \
        libmp                   .WAIT   \
        libnsl                  \
        libsecdb                .WAIT   \
        librpcsvc                       \
        libsocket               .WAIT   \
        libsctp                 \
        libsip                  \
        libcommputil            \
        libresolv                       \
        libresolv2              .WAIT   \
        libw                    .WAIT   \
        libintl                 .WAIT   \
        ../cmd/sgs/librtld_db   \
        libaio                  \
        libast                  \
        libdll                  \
        libcmd                  \
        libshell                \
        libsum                  \
        librt                   \
        libadm                  \
        libctf                  \
        libdtrace               \
        libdtrace_jni           \
        libcurses               \
        libtermcap              \
        libgen                  \
        libgss                  \
        libpam                  \
        libuuid                 \
        libthread               \
        libpthread      .WAIT   \
        libslp          \
        libbsdmalloc    \
        libdoor         \
        libdevinfo      \
        libdladm        \
        libdlpi         \
        libeti          \
        libcrypt        \
        libdns_sd       \
        libefi          \
        libfstyp        \
        libwanboot      \
        libwanbootutil  \
        libcryptoutil   \
        libinetutil     \
        libipadm        \
        libipmp         \
        libiscsit       \
        libkmf          \
        libkstat        \
        libkvm          \
        liblm           \
        libmalloc       \
        libmapmalloc    \
        libmtmalloc     \
        libnls          \
```

```
118         libnwam         \
119         libsmbios       \
120         libtecla        \
121         libumem         \
122         libnvpair       .WAIT   \
123         libexacct       \
124         libsasl         \
125         libldap5        \
126         libsldap        .WAIT   \
127         libbsm          \
128         libsys          \
129         libsysevent     \
130         libnisdb        \
131         libpool         \
132         libpp           \
133         libproc         \
134         libproject      \
135         libsendfile     \
136         nametoaddr      \
137         ncad_addr       \
138         hbaapi          \
139         smhba           \
140         sun_fc          \
141         sun_sas         \
142         gss_mechs/mech_krb5     .WAIT   \
143         libkrb5 .WAIT   \
144         krb5    .WAIT   \
145         libsmbfs        \
146         libfcoe         \
147         libsrpt         \
148         libstmf         \
149         libstmfproxy    \
150         libnsctl        \
151         libunistat      \
152         libdscfg        \
153         librdc          \
154         libinstzones    \
155         libpkg          \
156         libpcidb

158 SUBDIRS += \
159         passwdutil      \
160         pam_modules     \
161         crypt_modules   \
162         libadt_jni      \
163         abi             \
164         auditd_plugins  \
165         libvolmgt       \
166         libdevice       \
167         libdevid        \
168         libdhcpsvc      \
169         libc_db         \
170         libndmp         \
171         libsec          \
172         libtnfprobe     \
173         libtnf          \
174         libtnfctl       \
175         libdhcpagent    \
176         libdhcpdu       \
177         libdhcputil     \
178         libxnet         \
179         libipsecutil    \
186         libipsecutil
187 $(CLOSED_BUILD)SUBDIRS += \
188         $(CLOSED)/lib/libike
189 SUBDIRS += \
```



```
180         nsswitch        \
181         print           \
182         libuutil        \
183         libscf          \
184         libinetsvc      \
185         librestart      \
186         libsched        \
187         libelfsign      \
188         pkcs11          .WAIT   \
189         libpctx         .WAIT   \
190         libcpc          \
191         getloginx       \
192         watchmalloc     \
193         extendedFILE    \
194         madv            \
195         mpss            \
196         libdisasm       \
197         libwrap         \
198         libxcurses      \
199         libxcurses2     \
200         libbrand        .WAIT   \
201         libzonecfg      \
202         libzoneinfo     \
203         libzonestat     \
204         libtsnet        \
205         libtsol         \
206         gss_mechs/mech_spnego   \
207         gss_mechs/mech_dummy    \
208         gss_mechs/mech_dh       \
209         rpcsec_gss      \
210         libraidcfg      .WAIT   \
211         librcm          .WAIT   \
212         libcfgadm       .WAIT   \
213         libpicl         .WAIT   \
214         libpicltree     .WAIT \
215         raidcfg_plugins \
216         cfgadm_plugins  \
217         libmail         \
218         lvm             \
219         libsmedia       \
220         libipp          \
221         libdiskmgt      \
222         liblgrp         \
223         libfsmgt        \
224         fm              \
225         libavl          \
226         libcmdutils     \
227         libcontract     \
228         ../cmd/sendmail/libmilter       \
229         sasl_plugins    \
230         udapl           \
231         libzpool        \
232         libzfs_core     \
233         libzfs          \
234         libbe           \
235         pylibbe         \
236         libzfs_jni      \
237         pyzfs           \
238         pysolaris       \
239         libmapid        \
240         brand           \
241         policykit       \
242         hal             \
243         libshare        \
244         libsqlite       \
245         libidmap        \
```

```
246         libadutils      \
247         libipmi         \
248         libexacct/demo  \
249         libvrrpadm      \
250         libvscan        \
251         libgrubmgmt     \
252         smbsrv          \
253         libilb          \
254         scsi            \
255         libima          \
256         libsun_ima      \
257         mpapi           \
258         librstp         \
259         libreparse      \
260         libhotplug      \
261         libfruutils     .WAIT   \
262         libfru          \
263         $($(MACH)_SUBDIRS)

265 i386_SUBDIRS=           \
266         libntfs         \
267         libparted       \
268         libfdisk        \
269         libsaveargs

271 sparc_SUBDIRS= .WAIT    \
272         efcode          \
273         libds           \
274         libdscp         \
275         libprtdiag      .WAIT   \
276         libprtdiag_psr  \
277         libpri          \
278         librsc          \
279         storage         \
280         libpcp          \
281         libtsalarm      \
282         libv12n

284 FM_sparc_DEPLIBS= libpri

286 fm:                     \
287         libexacct       \
288         libipmi         \
289         libzfs          \
290         scsi            \
291         $(FM_$(MACH)_DEPLIBS)

293 #
294 # Create a special version of $(SUBDIRS) with no .WAIT's, for use with the
295 # clean and clobber targets (for more information, see those targets, below).
296 #
297 NOWAIT_SUBDIRS= $(SUBDIRS:.WAIT=)

299 DCSUBDIRS =             \
300         lvm

302 MSGSUBDIRS=             \
303         abi             \
304         auditd_plugins  \
305         brand           \
306         cfgadm_plugins  \
307         gss_mechs/mech_dh       \
308         gss_mechs/mech_krb5     \
309         krb5            \
310         libast          \
311         libbsm          \
```



```
312         libc            \
313         libcfgadm       \
314         libcmd          \
315         libcontract     \
316         libcurses       \
317         libdhcpsvc      \
318         libdhcputil     \
319         libipsecutil    \
320         libdiskmgt      \
321         libdladm        \
322         libdll          \
323         libgrubmgmt     \
324         libgss          \
325         libidmap        \
326         libipmp         \
327         libilb          \
328         libinetutil     \
329         libinstzones    \
330         libipadm        \
331         libnsl          \
332         libnwam         \
333         libpam          \
334         libpicl         \
335         libpool         \
336         libpkg          \
337         libpp           \
338         libscf          \
339         libsasl         \
340         libldap5        \
341         libsecdb        \
342         libshare        \
343         libshell        \
344         libsldap        \
345         libslp          \
346         libsmbfs        \
347         libsmedia       \
348         libsum          \
349         libtsol         \
350         libuutil        \
351         libvrrpadm      \
352         libvscan        \
353         libwanboot      \
354         libwanbootutil  \
355         libzfs          \
356         libzonecfg      \
357         lvm             \
358         madv            \
359         mpss            \
360         pam_modules     \
361         pyzfs           \
362         pysolaris       \
363         rpcsec_gss      \
364         libreparse
365 MSGSUBDIRS += \
366         $($(MACH)_MSGSUBDIRS)

368 sparc_MSGSUBDIRS=       \
369         libprtdiag      \
370         libprtdiag_psr

372 i386_MSGSUBDIRS= libfdisk

374 HDRSUBDIRS=             \
375         auditd_plugins  \
376         libast          \
377         libbrand        \
```

```
378         libbsm          \
379         libc            \
380         libcmd          \
381         libcmdutils     \
382         libcommputil    \
383         libcontract     \
384         libcpc          \
385         libctf          \
386         libcurses       \
387         libtermcap      \
388         libcryptoutil   \
389         libdevice       \
390         libdevid        \
391         libdevinfo      \
392         libdiskmgt      \
393         libdladm        \
394         libdll          \
395         libdlpi         \
396         libdhcpagent    \
397         libdhcpsvc      \
398         libdhcputil     \
399         libdisasm       \
400         libdns_sd       \
401         libdscfg        \
402         libdtrace       \
403         libdtrace_jni   \
404         libelfsign      \
405         libeti          \
406         libfru          \
407         libfstyp        \
408         libgen          \
409         libipadm        \
410         libipsecutil    \
411         libinetsvc      \
412         libinetutil     \
413         libinstzones    \
414         libipmi         \
415         libipmp         \
416         libipp          \
417         libiscsit       \
418         libkstat        \
419         libkvm          \
420         libmail         \
421         libmd           \
422         libmtmalloc     \
423         libndmp         \
424         libnvpair       \
425         libnsctl        \
426         libnsl          \
427         libnwam         \
428         libpam          \
429         libpcidb        \
430         libpctx         \
431         libpicl         \
432         libpicltree     \
433         libpool         \
434         libpp           \
435         libproc         \
436         libraidcfg      \
437         librcm          \
438         librdc          \
439         libscf          \
440         libsip          \
441         libsmbios       \
442         librestart      \
443         librpcsvc       \
```



```
444         librsm          \
445         librstp         \
446         libsasl         \
447         libsec          \
448         libshell        \
449         libslp          \
450         libsmedia       \
451         libsocket       \
452         libsqlite       \
453         libfcoe         \
454         libsrpt         \
455         libstmf         \
456         libstmfproxy    \
457         libsum          \
458         libsysevent     \
459         libtecla        \
460         libtnf          \
461         libtnfctl       \
462         libtnfprobe     \
463         libtsnet        \
464         libtsol         \
465         libvrrpadm      \
466         libvolmgt       \
467         libumem         \
468         libunistat      \
469         libuutil        \
470         libwanboot      \
471         libwanbootutil  \
472         libwrap         \
473         libxcurses2     \
474         libzfs          \
475         libzfs_core     \
476         libzfs_jni      \
477         libzoneinfo     \
478         libzonestat     \
479         hal             \
480         policykit       \
481         lvm             \
482         pkcs11          \
483         passwdutil      \
484         ../cmd/sendmail/libmilter       \
485         fm              \
486         udapl           \
487         libmapid        \
488         libkrb5         \
489         libsmbfs        \
490         libshare        \
491         libidmap        \
492         libvscan        \
493         libgrubmgmt     \
494         smbsrv          \
495         libilb          \
496         scsi            \
497         hbaapi          \
498         smhba           \
499         libima          \
500         libsun_ima      \
501         mpapi           \
502         libreparse      \
503         $($(MACH)_HDRSUBDIRS)

505 i386_HDRSUBDIRS=        \
506         libparted       \
```

```
515 $(CLOSED_BUILD)HDRSUBDIRS += \
516         $(CLOSED)/lib/libike
```

```
507         libfdisk        \
508         libsaveargs

510 sparc_HDRSUBDIRS=       \
511         libds           \
512         libdscp         \
513         libpri          \
514         libv12n         \
515         storage

517 all :=          TARGET= all
518 check :=        TARGET= check
519 clean :=        TARGET= clean
520 clobber :=      TARGET= clobber
521 install :=      TARGET= install
522 install_h :=    TARGET= install_h
523 lint :=         TARGET= lint
524 _dc :=          TARGET= _dc
525 _msg :=         TARGET= _msg

527 .KEEP_STATE:

529 #
530 # For the all and install targets, we clearly must respect library
531 # dependencies so that the libraries link correctly.  However, for
532 # the remaining targets (check, clean, clobber, install_h, lint, _dc
533 # and _msg), libraries do not have any dependencies on one another
534 # and thus respecting dependencies just slows down the build.
535 # As such, for these rules, we use pattern replacement to explicitly
536 # avoid triggering the dependency information.  Note that for clean,
537 # clobber and lint, we must use $(NOWAIT_SUBDIRS) rather than
538 # $(SUBDIRS), to prevent '.WAIT' from expanding to '.WAIT-nodepend'.
539 #

541 all:                    $(SUBDIRS)

543 install:                $(SUBDIRS) .WAIT install_extra

545 # extra libraries kept in other source areas
546 install_extra:
547         @cd ../cmd/sgs; pwd; $(MAKE) install_lib
548         @pwd

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

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

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

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

558 #
559 # Library interdependencies are called out explicitly here
560 #
561 auditd_plugins: libbsm libnsl libsecdb
562 gss_mechs/mech_krb5:    libgss libnsl libsocket libresolv pkcs11
563 libadt_jni:     libbsm
564 libast:         libsocket
565 libadutils:     libldap5 libresolv libsocket libnsl
566 nsswitch:       libadutils libidmap
567 libbe:          libzfs
568 libbsm:         libtsol
569 libcmd:         libsum libast libsocket libnsl
570 libcmdutils:    libavl
571 libcontract:    libnvpair
572 libdevid:       libdevinfo
```



```
573 libdevinfo:     libnvpair libsec
574 libdhcpagent:   libsocket libdhcputil libuuid libdlpi libcontract
575 libdhcpsvc:     libinetutil
576 libdhcputil:    libnsl libgen libinetutil libdlpi
577 libdladm:       libdevinfo libinetutil libsocket libscf librcm libnvpair \
578                 libexacct libnsl libkstat libcurses
579 libdll:         libast
580 libdlpi:        libinetutil libdladm
581 libds:          libsysevent
582 libdscfg:       libnsctl libunistat libsocket libnsl
583 libdtrace:      libproc libgen libctf
584 libdtrace_jni:  libuutil libdtrace
585 libefi:         libuuid
586 libfstyp:       libnvpair
587 libelfsign:     libcryptoutil libkmf
588 libidmap:       libadutils libldap5 libavl libsldap libuutil
589 libipadm:       libnsl libinetutil libsocket libdlpi libnvpair libdhcpagent \
590                 libdladm libsecdb
591 libiscsit:      libc libnvpair libstmf libuuid libnsl
592 libkmf:         libcryptoutil pkcs11
593 libnsl:         libmd5
594 libmapid:       libresolv
595 librdc:         libsocket libnsl libnsctl libunistat libdscfg
596 libuuid:        libdlpi
610 $(CLOSED_BUILD)libike:  libipsecutil libxnet libcryptoutil
597 libinetutil:    libsocket
598 libipsecutil:   libtecla libsocket
599 libinstzones:   libzonecfg libcontract
600 libpkg:         libwanboot libscf libadm
601 libnwam:        libscf
602 libsecdb:       libnsl
603 libsasl:        libgss libsocket pkcs11 libmd
604 sasl_plugins:   pkcs11 libgss libsocket libsasl
605 libsctp:        libsocket
606 libshell:       libast libcmd libdll libsocket libsecdb
607 libsip:        libmd5
608 libsmbfs:       libcmdutils libsocket libnsl libkrb5
609 libsocket:      libnsl
610 libstmfproxy:   libstmf libsocket libnsl libpthread
611 libsum:         libast
612 libsysevent:    libsecdb
613 libldap5:       libsasl libsocket libnsl libmd
614 libsldap:       libldap5 libtsol libnsl libc libscf libresolv
615 libpool:        libnvpair libexacct
616 libpp:          libast
617 libzonecfg:     libc libsocket libnsl libuuid libnvpair libsysevent libsec \
618                 libbrand libpool libscf
619 libproc:        ../cmd/sgs/librtld_db ../cmd/sgs/libelf libctf libsaveargs
620 libproject:     libpool libproc libsecdb
621 libtermcap:     libcurses
622 libtsnet:       libnsl libtsol libsecdb
623 libwrap:        libnsl libsocket
624 libwanboot:     libnvpair libresolv libnsl libsocket libdevinfo libinetutil \
625                 libdhcputil
626 libwanbootutil: libnsl
627 pam_modules:    libproject passwdutil smbsrv
628 libscf:         libuutil libmd libgen libsmbios libnsl
629 libinetsvc:     libscf
630 librestart:     libuutil libscf
631 libsaveargs:    libdisasm
632 ../cmd/sgs/libdl:       ../cmd/sgs/libconv
633 ../cmd/sgs/libelf:      ../cmd/sgs/libconv
634 pkcs11:         libcryptoutil
635 print:          libldap5
636 udapl/udapl_tavor:      udapl/libdat
637 libzfs:         libdevid libgen libnvpair libuutil \
```

```
638                 libadm libavl libefi libidmap libmd libzfs_core
639 libzfs_core:    libnvpair
640 libzfs_jni:     libdiskmgt libnvpair libzfs
641 libzpool:       libavl libumem libnvpair libcmdutils
642 libsec:         libavl libidmap
643 brand:          libc libsocket
644 libshare:       libscf libzfs libuuid libfsmgt libsecdb libumem libsmbfs
645 libexacct/demo: libexacct libproject libsocket libnsl
646 libtsalarm:     libpcp
647 smbsrv:         libsocket libnsl libmd libxnet libpthread librt \
648                 libshare libidmap pkcs11 libsqlite libcryptoutil \
649                 libreparse libcmdutils
650 libv12n:        libds libuuid
651 libvrrpadm:     libsocket libdladm libscf
652 libvscan:       libscf
653 libfru:         libfruutils
654 scsi:           libnvpair libfru
655 mpapi:          libpthread libdevinfo libsysevent libnvpair
656 sun_fc:         libdevinfo libsysevent libnvpair
657 libsun_ima:     libdevinfo libsysevent libnsl
658 sun_sas:        libdevinfo libsysevent libnvpair libkstat libdevid
659 libgrubmgmt:    libdevinfo libzfs libfstyp
660 pylibbe:        libbe libzfs
661 pyzfs:          libnvpair libzfs
662 pysolaris:      libsec libidmap
663 libreparse:     libnvpair
664 libhotplug:     libnvpair
665 cfgadm_plugins: libhotplug
666 libilb:         libsocket
667 $(INTEL_BUILD)libdiskmgt:libfdisk

669 #
670 # The reason this rule checks for the existence of the
671 # Makefile is that some of the directories do not exist
672 # in certain situations (e.g., exportable source builds,
673 # OpenSolaris).
674 #
675 $(SUBDIRS): FRC
676         @if [ -f $@/Makefile  ]; then \
677                 cd $@; pwd; $(MAKE) $(TARGET); \
678         else \
679                 true; \
680         fi

682 $(SUBDIRS:%=%-nodepend):
683         @if [ -f $(@:%-nodepend=%)/Makefile  ]; then \
684                 cd $(@:%-nodepend=%); pwd; $(MAKE) $(TARGET); \
685         else \
686                 true; \
687         fi

689 FRC:
```

```
**********************************************************
    8584 Wed Oct 16 17:41:05 2013
new/usr/src/lib/Makefile.lib
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
#
#
# Definitions common to libraries.
#
# include global definitions; SRC should be defined in the shell.
# SRC is needed until RFE 1026993 is implemented.

include         $(SRC)/Makefile.master

LORDER=         lorder
TSORT=          tsort
AWK=            awk

#
# By default, we define the source directory for libraries to be
# one level up from the ISA-specific directory, where the code is
# actually built.  Many libraries define a 'common' directory to
# contain the source.  These libraries must redefine SRCDIR as:
#       SRCDIR = ../common
# Other variations are possible (../port, ../src, etc).
#
SRCDIR =        ..

#
# We define MAPFILES here for the benefit of most libraries, those that
# follow the convention of having source files and other common files
# in the $(SRCDIR) directory.  Libraries that do not follow this
# convention must define MAPFILES, or MAPFILEDIR for themselves.
# Libraries that do follow this convention but that need supplemental
# ISA-specific mapfiles can augment MAPFILES like this:
#       MAPFILES += mapfile-vers
#
MAPFILEDIR =    $(SRCDIR)
MAPFILES =      $(MAPFILEDIR)/mapfile-vers

#
# If HDRDIR is left unset, then it's possible for the $(ROOTHDRDIR)/%
```



```
# install rule in lib/Makefile.targ to generate false matches if there
# are any common directory names between / and /usr/include ('xfn' is
# one common example).  To prevent this, we set HDRDIR to a directory
# name that will almost surely not exist on the build machine.
#
HDRDIR=         /__nonexistent_directory__

#
# We don't build archive (*.a) libraries by default anymore.
# If a component of the build needs to build an archive library
# for its own internal purposes, it can define LIBS for itself
# after including Makefile.lib, like this:
#       LIBS = $(LIBRARY)
# or:
#       LIBS = $(LIBRARYCCC)
# Archive libraries must not be installed in the proto area.
#
LIBS=
MACHLIBS=       $(LIBS:%=$(MACH)/%)
MACHLIBS64=     $(LIBS:%=$(MACH64)/%)
DYNLIB=         $(LIBRARY:.a=.so$(VERS))
DYNLIBPSR=      $(LIBRARY:.a=_psr.so$(VERS))
DYNLIBCCC=      $(LIBRARYCCC:.a=.so$(VERS))
LIBLINKS=       $(LIBRARY:.a=.so)
LIBLINKSCCC=    $(LIBRARYCCC:.a=.so)
LIBNAME=        $(LIBRARY:lib%.a=%)
LIBLINKPATH=
LIBNULL=        null.a
ROOTHDRDIR=     $(ROOT)/usr/include
ROOTLIBDIR=     $(ROOT)/usr/lib
ROOTLIBDIR64=   $(ROOT)/usr/lib/$(MACH64)
ROOTFS_LIBDIR=  $(ROOT)/lib
ROOTFS_LIBDIR64=        $(ROOT)/lib/$(MACH64)
ROOTLINTDIR=    $(ROOTLIBDIR)
ROOTFS_LINTDIR= $(ROOTFS_LIBDIR)
ROOTFS_LINTDIR64=       $(ROOTFS_LIBDIR64)
ROOTHDRS=       $(HDRS:%=$(ROOTHDRDIR)/%)
HDRSRCS=        $(HDRS:%=$(HDRDIR)/%)
CHECKHDRS=      $(HDRSRCS:%.h=%.check)
ROOTLIBS=       $(LIBS:%=$(ROOTLIBDIR)/%)
ROOTLIBS64=     $(LIBS:%=$(ROOTLIBDIR64)/%)
ROOTFS_LIBS=    $(DYNLIB:%=$(ROOTFS_LIBDIR)/%)
ROOTFS_LIBS64=  $(DYNLIB:%=$(ROOTFS_LIBDIR64)/%)
ROOTLINKS=      $(ROOTLIBDIR)/$(LIBLINKS)
ROOTLINKS64=    $(ROOTLIBDIR64)/$(LIBLINKS)
ROOTFS_LINKS=   $(ROOTFS_LIBDIR)/$(LIBLINKS)
ROOTFS_LINKS64= $(ROOTFS_LIBDIR64)/$(LIBLINKS)
ROOTLINKSCCC=   $(ROOTLIBDIR)/$(LIBLINKSCCC)
ROOTLINKSCCC64= $(ROOTLIBDIR64)/$(LIBLINKSCCC)
ROOTFS_LINKSCCC=        $(ROOTFS_LIBDIR)/$(LIBLINKSCCC)
ROOTFS_LINKSCCC64=      $(ROOTFS_LIBDIR64)/$(LIBLINKSCCC)
ROOTLINT=       $(LINTSRC:%=$(ROOTLINTDIR)/%)
ROOTFS_LINT=    $(LINTSRC:%=$(ROOTFS_LINTDIR)/%)
ROOTFS_LINT64=  $(LINTSRC:%=$(ROOTFS_LINTDIR64)/%)
ROOTMAN3=     $(ROOT)/usr/share/man/man3
ROOTMAN3FILES=        $(MAN3FILES:%=$(ROOTMAN3)/%)
$(ROOTMAN3FILES) := FILEMODE= 444

# Demo rules
DEMOFILES=
DEMOFILESRCDIR=         common
ROOTDEMODIRBASE=        __nonexistent_directory__
ROOTDEMODIRS=
ROOTDEMOFILES=  $(DEMOFILES:%=$(ROOTDEMODIRBASE)/%)
$(ROOTDEMODIRS) :=      DIRMODE =       755
```

```
125 LINTLIB=        llib-l$(LIBNAME).ln
126 LINTFLAGS=      -uaxm
127 LINTFLAGS64=    -uaxm -m64
128 LINTSRC=        $(LINTLIB:%.ln=%)
129 LINTOUT=        lint.out
130 ARFLAGS=        r
131 SONAME=         $(DYNLIB)
132 # For most libraries, we should be able to resolve all symbols at link time,
133 # either within the library or as dependencies, all text should be pure, and
134 # combining relocations into one relocation table reduces startup costs.
135 # All options are tunable to allow overload/omission from lower makefiles.


138 HSONAME=        -h$(SONAME)
139 DYNFLAGS=       $(HSONAME) $(ZTEXT) $(ZDEFS) $(BDIRECT) \
140                 $(MAPFILES:%=-M%) $(MAPFILE.PGA:%=-M%) $(MAPFILE.NED:%=-M%)

142 LDLIBS=         $(LDLIBS.lib)

144 OBJS=           $(OBJECTS:%=objs/%)
145 PICS=           $(OBJECTS:%=pics/%)

147 # Declare that all library .o's can all be made in parallel.
148 # The DUMMY target is for those instances where OBJS and PICS
149 # are empty (to avoid an unconditional .PARALLEL declaration).
150 .PARALLEL:      $(OBJS) $(PICS) DUMMY

152 # default value for "portable" source
153 SRCS=           $(OBJECTS:%.o=$(SRCDIR)/%.c)

155 # default build of an archive and a shared object,
156 # overridden locally when extra processing is needed
157 BUILD.AR=       $(AR) $(ARFLAGS) $@ $(AROBJS)
158 BUILD.SO=       $(CC) -o $@ $(GSHARED) $(DYNFLAGS) $(PICS) $(EXTPICS) $(LDLIBS)
159 BUILDCCC.SO=    $(CCC) -o $@ $(GSHARED) $(DYNFLAGS) $(PICS) $(EXTPICS) $(LDLIBS)

161 # default dynamic library symlink
162 # IMPORTANT:: If you change INS.liblink OR INS.liblink64 here, then you
163 # MUST also change the corresponding override definitions in
164 # $CLOSED/Makefile.tonic.
165 #
166 # If you do not do this, then the closedbins build for the OpenSolaris
167 # community will break. PS, the gatekeepers will be upset too.
168 #
162 INS.liblink=    -$(RM) $@; $(SYMLINK) $(LIBLINKPATH)$(LIBLINKS)$(VERS) $@
163 INS.liblinkccc= -$(RM) $@; $(SYMLINK) $(LIBLINKPATH)$(LIBLINKSCCC)$(VERS) $@

165 # default 64-bit dynamic library symlink
166 INS.liblink64=  -$(RM) $@; $(SYMLINK) $(LIBLINKPATH)$(LIBLINKS)$(VERS) $@
167 INS.liblinkccc64= -$(RM) $@; $(SYMLINK) $(LIBLINKPATH)$(LIBLINKSCCC)$(VERS) $@

169 #
170 # If appropriate, augment POST_PROCESS_O and POST_PROCESS_SO to do CTF
171 # processing.  We'd like to just conditionally append to POST_PROCESS_O and
172 # POST_PROCESS_SO, but ParallelMake has a bug which causes the same value to
173 # sometimes get appended more than once, which will cause ctfconvert to fail.
174 # So, instead we introduce CTFCONVERT_POST and CTFMERGE_POST, which are always
175 # appended to POST_PROCESS_O and POST_PROCESS_SO but are no-ops unless CTF
176 # processing should be done.
177 #
178 CTFCONVERT_POST = :
179 CTFMERGE_POST   = :
180 POST_PROCESS_O += ; $(CTFCONVERT_POST)
181 POST_PROCESS_SO += ; $(CTFMERGE_POST)

183 CTFMERGE_LIB    = $(CTFMERGE) $(CTFMRGFLAGS) -t -f -L VERSION -o $@ $(PICS)
```



```
185 # conditional assignments

187 $(OBJS)  :=     sparc_CFLAGS += -xregs=no%appl

189 $(PICS)  :=     sparc_CFLAGS += -xregs=no%appl $(sparc_C_PICFLAGS)
190 $(PICS)  :=     sparcv9_CFLAGS += -xregs=no%appl $(sparcv9_C_PICFLAGS)
191 $(PICS)  :=     i386_CFLAGS += $(i386_C_PICFLAGS)
192 $(PICS)  :=     amd64_CFLAGS += $(amd64_C_PICFLAGS)
193 $(PICS)  :=     CCFLAGS += $(CC_PICFLAGS)
194 $(PICS)  :=     CPPFLAGS += -DPIC -D_REENTRANT
195 $(PICS)  :=     sparcv9_CCFLAGS += -xregs=no%appl $(sparcv9_CC_PICFLAGS)
196 $(PICS)  :=     amd64_CCFLAGS += $(amd64_CC_PICFLAGS)
197 $(PICS)  :=     CFLAGS += $(CTF_FLAGS)
198 $(PICS)  :=     CFLAGS64 += $(CTF_FLAGS)
199 $(PICS)  :=     CTFCONVERT_POST = $(CTFCONVERT_O)
200 $(DYNLIB) :=    CTFMERGE_POST = $(CTFMERGE_LIB)

202 $(LINTLIB):=    LOG = -DLOGGING
203 $(LIBRARY):=    AROBJS = $(OBJS)
204 $(LIBRARY):=    DIR = objs
205 $(DYNLIB):=     DIR = pics
206 $(DYNLIBCCC):=  DIR = pics

208 SONAMECCC=      $(DYNLIBCCC)
209 HSONAMECCC=     -h $(SONAMECCC)
210 #
211 # Keep in sync with the standard DYNFLAGS
212 #
213 $(DYNLIBCCC):=  DYNFLAGS = $(HSONAMECCC) $(ZTEXT) $(ZDEFS) \
214                 $(MAPFILES:%=-M%) $(MAPFILE.PGA:%=-M%) $(MAPFILE.NED:%=-M%) \
215                 $(BDIRECT) $(NORUNPATH)


218 # build rule for "portable" source
219 objs/%.o pics/%.o: %.c
220         $(COMPILE.c) -o $@ $<
221         $(POST_PROCESS_O)

223 objs/%.o pics/%.o: %.cc
224         $(COMPILE.cc) -o $@ $<
225         $(POST_PROCESS_O)

227 .PRECIOUS: $(LIBS)

229 # Define the majority text domain in this directory.
230 TEXT_DOMAIN= SUNW_OST_OSLIB

232 $(ROOTMAN3)/%: %.sunman
233         $(INS.rename)

235 #
236 # For library source code, we expect that some symbols may not be used or
237 # may *appear* to be able to rescoped to static; shut lint up.  Never add
238 # a flag here unless you're *sure* that all libraries need to be linted
239 # with it.
240 #
241 LINTCHECKFLAGS = -m -erroff=E_NAME_DEF_NOT_USED2
242 LINTCHECKFLAGS += -erroff=E_NAME_DECL_NOT_USED_DEF2

244 #
245 # Target Architecture
246 #
247 TARGETMACH=     $(MACH)

249 #
```

```
250 # Allow people to define their own clobber rules.  Normal makefiles
251 # shouldn't override this - they should override $(CLOBBERFILES) instead.
252 #
253 CLOBBERTARGFILES= $(LIBS) $(DYNLIB) $(CLOBBERFILES)
```

```
**********************************************************
    1268 Wed Oct 16 17:41:06 2013
new/usr/src/lib/raidcfg_plugins/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# ident "%Z%%M% %I%     %E% SMI"
#
# lib/raidcfg_plugins/Makefile
#

include $(SRC)/Makefile.master

CLOSED_PLUGIN = $(CLOSED)/lib/raidcfg_plugins/

$(CLOSED_BUILD)COMMON_SUBDIRS += $(CLOSED_PLUGIN)/mpt

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

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber
lint:=          TARGET= lint
_msg:=          TARGET= _msg

.KEEP_STATE:

all clean clobber lint install _msg: $(SUBDIRS)

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

FRC:
```

```
**********************************************************
    1614 Wed Oct 16 17:41:06 2013
new/usr/src/pkg/Makefile.lic
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
#

#
# PKGDEBUG and LICROOT should be defined on the command line or in
# the environment when using this Makefile, as they generally come
# from usr/src/pkg/Makefile.  The remaining env vars should already
# be part of a working build environment.
#

include $(SRC)/Makefile.master

#
# For license files in the open source tree, always copy them from src
# into the license staging directory in the proto area
#
$(LICROOT)/usr/src/%: $(CODEMGR_WS)/usr/src/%
        $(PKGDEBUG)if [ ! -d $(@D) ]; then mkdir -p $(@D); fi
        $(PKGDEBUG)$(INS.file)

#
# A build without closed source relies on closed-bins extraction to stage
# the license file and does nothing here.
# For license files in the closed source tree, the desired action
# depends on the type of build.
#
$(LICROOT)/usr/closed/%:
# 1. For a normal build, with closed source present, simply copy the
#    file into the license staging directory.
#
# 2. For a Tonic build, copy the file into both open and closed
#    license staging directories.
#
# 3. For a build without closed source, rely on closed-bins extraction
#    to stage the license file, and do nothing here.
#
```



```
$(TONICBUILD)INS= install -O
$(LICROOT)/usr/closed/%: $(CLOSED_BUILD) $(CODEMGR_WS)/usr/closed/%
        $(CLOSED_BUILD)$(PKGDEBUG) \
        if [ ! -d $(@D) ]; then \
                mkdir -p $(@D); \
        fi; \
        $(TONICBUILD) $(RM) $(@:$(ROOT)/%=$(CLOSEDROOT)/%)
        $(CLOSED_BUILD)$(PKGDEBUG)$(INS.file)
```

```
**********************************************************
    1957 Wed Oct 16 17:41:06 2013
new/usr/src/psm/stand/boot/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

include         ../../../Makefile.master

sparcv9_ARCHITECTURES = sparcv9
sparc_ARCHITECTURES = $(sparcv9_ARCHITECTURES)

SUBDIRS = $($(MACH)_ARCHITECTURES)

all             :=      TARGET= all
install         :=      TARGET= install
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint

.KEEP_STATE:

all install lint clean: $(SUBDIRS)

clobber: $(SUBDIRS)
        $(RM) make.out lint.out

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

#
# Cross-reference customization: include all boot-related source files.
#
UTSDIR =        ../../../uts
UTSCLOSED =     ../../../../closed/uts
STANDLIBDIR =   ../../../stand/lib
STANDSYSDIRS =  ../../../stand/sys
PROMDIRS =      ../../promif
NAMESDIRS =     ../lib/names
XRDIRS +=       $(STANDLIBDIR) $(STANDSYSDIRS) $(PROMDIRS) $(NAMESDIRS)
```



```
#
# Components beginning with B! are in the open and closed trees; those
# beginning with O! are just in the open tree.
#
XRINCCOMP = B!sun4u O!sfmmu O!sparc/v7 O!sparc/v9 B!sparc B!sun B!common
XRINC_TMP = $(XRINCCOMP:B!%=$(UTSDIR)/%)
XRINCDIRS = $(XRINC_TMP:O!%=$(UTSDIR)/%)
$(CLOSED_BUILD)XRINC_TMP = $(XRINCCOMP:B!%=$(UTSDIR)/% $(UTSCLOSED)/%)
$(CLOSED_BUILD)XRINCDIRS = $(XRINC_TMP:O!%=$(UTSDIR)/%)

cscope.out tags: FRC
        $(XREF) -x $@

FRC:
```

```
**********************************************************
    8013 Wed Oct 16 17:41:06 2013
new/usr/src/tools/env/developer.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
#

#       Configuration variables for the runtime environment of the nightly
# build script and other tools for construction and packaging of releases.
# This script is sourced by 'nightly' and 'bldenv' to set up the environment
# for the build. This example is suitable for building a developers workspace,
# which will contain the resulting packages and archives. It is based off
# the onnv release. It sets NIGHTLY_OPTIONS to make nightly do:
#       check ELF ABI/versioning (-A)
#       runs 'make check' (-C)
#       DEBUG and non-DEBUG builds (-D)
#       runs lint in usr/src (-l plus the LINTDIRS variable)
#       sends mail on completion (-m and the MAILTO variable)
#       creates packages for PIT/RE (-p)
#       checks for changes in ELF runpaths (-r)
#
NIGHTLY_OPTIONS="-ACDlmpr";             export NIGHTLY_OPTIONS

# This is a variable for the rest of the script - GATE doesn't matter to
# nightly itself
GATE=onnv-bugfixes;                     export GATE

# CODEMGR_WS - where is your workspace at (or what should nightly name it)
CODEMGR_WS="/builds/$GATE";                     export CODEMGR_WS

# PARENT_WS is used to determine the parent of this workspace. This is
# for the options that deal with the parent workspace (such as where the
# proto area will go).
#
# If you use this, it must be local (or nfs): nightly cannot copy
# over ssh or http.
PARENT_WS="/ws/onnv-gate";                      export PARENT_WS

# CLONE_WS is the workspace nightly should do a bringover from.
CLONE_WS="ssh://anonhg@onnv.sfbay.sun.com//export/onnv-clone";  export CLONE_WS
```



```
# CLOSED_CLONE_WS is the workspace from which nightly should acquire
# the usr/closed tree.
CLOSED_CLONE_WS="${CLONE_WS}/usr/closed";       export CLOSED_CLONE_WS

# This flag controls whether to build the closed source.  If
# undefined, nightly(1) and bldenv(1) will set it according to whether
# the closed source tree is present.  CLOSED_IS_PRESENT="no" means not
# building the closed sources.
# CLOSED_IS_PRESENT="yes";              export CLOSED_IS_PRESENT

# The bringover, if any, is done as STAFFER.
# Set STAFFER to your own login as gatekeeper or developer
# The point is to use group "staff" and avoid referencing the parent
# workspace as root.
# Some scripts optionally send mail messages to MAILTO.
#
STAFFER=nobody;                         export STAFFER
MAILTO=$STAFFER;                        export MAILTO

# The project (see project(4)) under which to run this build.  If not
# specified, the build is simply run in a new task in the current project.
BUILD_PROJECT=;                         export BUILD_PROJECT

# You should not need to change the next four lines
LOCKNAME="`basename $CODEMGR_WS`_nightly.lock"; export LOCKNAME
ATLOG="$CODEMGR_WS/log";                        export ATLOG
LOGFILE="$ATLOG/nightly.log";                   export LOGFILE
MACH=`uname -p`;                                export MACH

# When the -A flag is specified, and ELF_DATA_BASELINE_DIR is defined,
# the ELF interface description file resulting from the build is compared
# to that from the specified directory. This ensures that our object
# versioning evolves in a backward compatible manner.
#
# You should not need to change this unless you wish to use locally cached
# baseline files. If you use this, it must be local (or nfs): nightly cannot
# copy over ssh or http.
#
ELF_DATA_BASELINE_DIR="/ws/onnv-gate/usr/src/ELF-data-baseline.$MACH";  export E

# This is usually just needed if the closed tree is missing, or when
# building a project gate with the -O (cap oh) flag.
# ON_CRYPTO_BINS="$PARENT_WS/packages/$MACH/on-crypto.$MACH.tar.bz2"
# export ON_CRYPTO_BINS

# REF_PROTO_LIST - for comparing the list of stuff in your proto area
# with. Generally this should be left alone, since you want to see differences
# from your parent (the gate).
#
REF_PROTO_LIST=$PARENT_WS/usr/src/proto_list_${MACH}; export REF_PROTO_LIST

#
#       build environment variables, including version info for mcs, motd,
# motd, uname and boot messages. Mostly you shouldn't change this except
# when the release slips (nah) or you move an environment file to a new
# release
#
ROOT="$CODEMGR_WS/proto/root_${MACH}";  export ROOT
SRC="$CODEMGR_WS/usr/src";              export SRC
VERSION="$GATE";                        export VERSION

#
# the RELEASE and RELEASE_DATE variables are set in Makefile.master;
# there might be special reasons to override them here, but that
# should not be the case in general
```

```
119 #
120 # RELEASE="5.10.1";                     export RELEASE
121 # RELEASE_DATE="October 2007";          export RELEASE_DATE

123 # proto area in parent for optionally depositing a copy of headers and
124 # libraries corresponding to the protolibs target
125 # not applicable given the NIGHTLY_OPTIONS
126 #
127 PARENT_ROOT=$PARENT_WS/proto/root_$MACH; export PARENT_ROOT
128 PARENT_TOOLS_ROOT=$PARENT_WS/usr/src/tools/proto/root_$MACH-nd; export PARENT_TO

130 #
131 # Package creation variables.  You probably shouldn't change these,
132 # either.
133 #
134 # PKGARCHIVE determines where repositories will be created.
135 #
136 # PKGPUBLISHER* control the publisher settings for those repositories.
137 #
138 PKGARCHIVE="${CODEMGR_WS}/packages/${MACH}/nightly";    export PKGARCHIVE
139 # PKGPUBLISHER_REDIST="on-redist";                      export PKGPUBLISHER_REDI
140 # PKGPUBLISHER_NONREDIST="on-extra";                    export PKGPUBLISHER_NONR

142 # we want make to do as much as it can, just in case there's more than
143 # one problem.
144 MAKEFLAGS=k;    export MAKEFLAGS

146 # Magic variable to prevent the devpro compilers/teamware from sending
147 # mail back to devpro on every use.
148 UT_NO_USAGE_TRACKING="1"; export UT_NO_USAGE_TRACKING

150 # Build tools - don't set these unless you know what you're doing.  These
151 # variables allows you to get the compilers and onbld files locally or
152 # through cachefs.  Set BUILD_TOOLS to pull everything from one location.
153 # Alternately, you can set ONBLD_TOOLS to where you keep the contents of
154 # SUNWonbld and SPRO_ROOT to where you keep the compilers.
155 #
156 #BUILD_TOOLS=/opt;                              export BUILD_TOOLS
157 #ONBLD_TOOLS=/opt/onbld;                        export ONBLD_TOOLS
158 #SPRO_ROOT=/opt/SUNWspro;                       export SPRO_ROOT

160 # This goes along with lint - it is a series of the form "A [y|n]" which
161 # means "go to directory A and run 'make lint'" Then mail me (y) the
162 # difference in the lint output. 'y' should only be used if the area you're
163 # linting is actually lint clean or you'll get lots of mail.
164 # You shouldn't need to change this though.
165 #LINTDIRS="$SRC y";     export LINTDIRS

167 #
168 # Reference to IA32 IHV workspace, proto area and packages
169 #
170 #IA32_IHV_WS=/ws/${GATE}-ihv;                           export IA32_IHV_WS
171 #IA32_IHV_ROOT=$IA32_IHV_WS/proto/root_i386;            export IA32_IHV_ROOT
172 #IA32_IHV_PKGS=$IA32_IHV_WS/packages/i386/nightly;      export IA32_IHV_PKGS

174 #
175 # Reference to binary-only IA32 IHV packages
176 #
177 #IA32_IHV_BINARY_PKGS=/ws/${GATE}-ihv-bin
178 #export IA32_IHV_BINARY_PKGS

180 # Set this flag to 'n' to disable the automatic validation of the dmake
181 # version in use.  The default is to check it.
182 #CHECK_DMAKE=y

184 # Set this flag to 'n' to disable the use of 'checkpaths'.  The default,
```



```
185 # if the 'N' option is not specified, is to run this test.
186 #CHECK_PATHS=y

188 # Set this flag to 'y' to enable the use of elfsigncmp to validate the
189 # output of elfsign.  Doing so requires that 't' be set in NIGHTLY_OPTIONS.
190 # The default is to not verify them.
191 #VERIFY_ELFSIGN=n

193 # BRINGOVER_FILES is the list of files nightly passes to bringover.
194 # If not set the default is "usr", but it can be used for bringing
195 # over deleted_files or other nifty directories.
196 #BRINGOVER_FILES="usr deleted_files"

198 # POST_NIGHTLY can be any command to be run at the end of nightly.  See
199 # nightly(1) for interactions between environment variables and this command.
200 #POST_NIGHTLY=
```

```
**********************************************************
    8626 Wed Oct 16 17:41:06 2013
new/usr/src/tools/env/gatekeeper.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
#

#       Configuration variables for the runtime environment of the nightly
# build script and other tools for construction and packaging of releases.
# This script is sourced by 'nightly' and 'bldenv' to set up the environment
# for the build. This example is suitable for building a gate,
# which will contain the resulting packages and archives (builds of the gate
# are done in children and then the resulting archives, packages, and proto
# area are put into the parent for everyone to use). It is based off
# the onnv release. It sets NIGHTLY_OPTIONS to make nightly do:
#       DEBUG and non-DEBUG builds (-D)
#       creates packages for PIT/RE (-p)
#       checks for new interfaces in libraries (-A)
#       runs 'make check' (-C)
#       runs lint in usr/src (-l plus the LINTDIRS variable)
#       sends mail on completion (-m and the MAILTO variable)
#       updates the protolist in the parent for children to compare with (-u)
#       updates the proto area in the parent when done (-U)
#       checks for changes in ELF runpaths (-r)
#       checks for changes in unreferenced files (-f)
#
NIGHTLY_OPTIONS="-ADClmpuUrf";          export NIGHTLY_OPTIONS

# This is a variable for the rest of the script - GATE doesn't matter to
# nightly itself
GATE=onnv-gate;                                 export GATE

# CODEMGR_WS - where is your workspace at (or what should nightly name it)
# there is only one definition here, which assumes all the gate build machines
# (sparc and x86) are set up the same. But remember, this is a script, so
# you _could_ look at $MACH or 'uname -n' and set these variables differently.
CODEMGR_WS="/builds/$GATE";                     export CODEMGR_WS

# PARENT_WS is used to determine the parent of this workspace. This is
# for the options that deal with the parent workspace (such as where the
```



```
# proto area will go).
#
# If you use this, it must be local (or nfs): nightly cannot copy
# over ssh or http.
PARENT_WS="/ws/$GATE";                          export PARENT_WS

# CLONE_WS is the workspace nightly should do a bringover from.
CLONE_WS="ssh://anonhg@onnv.sfbay.sun.com//export/onnv-clone";  export CLONE_WS

# CLOSED_CLONE_WS is the workspace from which nightly will acquire the
# usr/closed tree.
CLOSED_CLONE_WS="${CLONE_WS}/usr/closed"
export CLOSED_CLONE_WS

# This flag controls whether to build the closed source.  If
# undefined, nightly(1) and bldenv(1) will set it according to whether
# the closed source tree is present.  CLOSED_IS_PRESENT="no" means not
# building the closed sources.
# CLOSED_IS_PRESENT="yes";              export CLOSED_IS_PRESENT

# The bringover, if any, is done as STAFFER.
# Set STAFFER to your own login as gatekeeper or integration engineer.
# The point is to use group "staff" and avoid referencing the parent
# workspace as root.
# Some scripts optionally send mail messages to MAILTO.
#
STAFFER=nobody;                         export STAFFER
MAILTO=$STAFFER;                        export MAILTO

# The project (see project(4)) under which to run this build.  If not
# specified, the build is simply run in a new task in the current project.
BUILD_PROJECT=;                         export BUILD_PROJECT

# You should not need to change the next four lines
LOCKNAME="`basename $CODEMGR_WS`_nightly.lock"; export LOCKNAME
ATLOG="$CODEMGR_WS/log";                        export ATLOG
LOGFILE="$ATLOG/nightly.log";                   export LOGFILE
MACH=`uname -p`;                                export MACH

# When the -A flag is specified, and ELF_DATA_BASELINE_DIR is defined,
# the ELF interface description file resulting from the build is compared
# to that from the specified directory. This ensures that our object
# versioning evolves in a backward compatible manner.
#
# You should not need to change this unless you wish to use locally cached
# baseline files. If you use this, it must be local (or nfs): nightly cannot
# copy over ssh or http.
#
ELF_DATA_BASELINE_DIR="/ws/onnv-gate/usr/src/ELF-data-baseline.$MACH";  export E

# This is usually just needed if the closed tree is missing, or when
# building a project gate with the -O (cap oh) flag.
# ON_CRYPTO_BINS="$PARENT_WS/packages/$MACH/on-crypto.$MACH.tar.bz2"
# export ON_CRYPTO_BINS

# REF_PROTO_LIST - for comparing the list of stuff in your proto area
# with. Generally this should be left alone, since you want to see differences
# between todays build and yesterdays.
#
REF_PROTO_LIST=$PARENT_WS/usr/src/proto_list_${MACH}; export REF_PROTO_LIST

#
#       build environment variables, including version info for mcs, motd,
# motd, uname and boot messages. Mostly you shouldn't change this except
# when the release slips (nah) or when starting a new release.
#
```

```
119 ROOT="$CODEMGR_WS/proto/root_${MACH}";  export ROOT
120 SRC="$CODEMGR_WS/usr/src";              export SRC
121 VERSION="$GATE";                        export VERSION

123 #
124 # the RELEASE and RELEASE_DATE variables are set in Makefile.master;
125 # there might be special reasons to override them here, but that
126 # should not be the case in general
127 #
128 # RELEASE="5.10.1";                     export RELEASE
129 # RELEASE_DATE="October 2007";          export RELEASE_DATE

131 # proto area in parent for optionally depositing a copy of headers and
132 # libraries corresponding to the protolibs target
133 #
134 PARENT_ROOT=$PARENT_WS/proto/root_$MACH; export PARENT_ROOT
135 PARENT_TOOLS_ROOT=$PARENT_WS/usr/src/tools/proto/root_$MACH-nd; export PARENT_TO

137 #
138 # Package creation variables.  You probably shouldn't change these,
139 # either.
140 #
141 # PKGARCHIVE determines where repositories will be created.
142 #
143 # PKGPUBLISHER* control the publisher settings for those repositories.
144 #
145 PKGARCHIVE="${PARENT_WS}/packages/${MACH}/nightly";     export PKGARCHIVE
146 # PKGPUBLISHER_REDIST="on-nightly";                     export PKGPUBLISHER_REDI
147 # PKGPUBLISHER_NONREDIST="on-extra";                    export PKGPUBLISHER_NONR


150 # we want make to do as much as it can, just in case there's more than
151 # one problem. This is especially important with the gate, since multiple
152 # unrelated broken things can be integrated.
153 MAKEFLAGS=k;    export MAKEFLAGS

155 # Magic variable to prevent the devpro compilers/teamware from sending
156 # mail back to devpro on every use.
157 UT_NO_USAGE_TRACKING="1"; export UT_NO_USAGE_TRACKING

159 # Build tools - don't set these unless you know what you're doing.  These
160 # variables allows you to get the compilers and onbld files locally or
161 # through cachefs.  Set BUILD_TOOLS to pull everything from one location.
162 # Alternately, you can set ONBLD_TOOLS to where you keep the contents of
163 # SUNWonbld and SPRO_ROOT to where you keep the compilers.
164 #
165 #BUILD_TOOLS=/opt;                              export BUILD_TOOLS
166 #ONBLD_TOOLS=/opt/onbld;                        export ONBLD_TOOLS
167 #SPRO_ROOT=/opt/SUNspro;                        export SPRO_ROOT

169 # This goes along with lint - it is a series of the form "A [y|n]" which
170 # means "go to directory A and run 'make lint'" Then mail me (y) the
171 # difference in the lint output. 'y' should only be used if the area you're
172 # linting is actually lint clean or you'll get lots of mail.
173 # You shouldn't need to change this though.
174 #LINTDIRS="$SRC y";     export LINTDIRS

176 #
177 # Reference to IA32 IHV workspace, proto area and packages
178 #
179 #IA32_IHV_WS=/ws/${GATE}-ihv;                           export IA32_IHV_WS
180 #IA32_IHV_ROOT=$IA32_IHV_WS/proto/root_i386;            export IA32_IHV_ROOT
181 #IA32_IHV_PKGS=$IA32_IHV_WS/packages/i386/nightly;      export IA32_IHV_PKGS

183 #
184 # Reference to binary-only IA32 IHV packages
```



```
185 #
186 #IA32_IHV_BINARY_PKGS=/ws/${GATE}-ihv-bin
187 #export IA32_IHV_BINARY_PKGS

189 # Set this flag to 'n' to disable the automatic validation of the dmake
190 # version in use.  The default is to check it.
191 #CHECK_DMAKE=y

193 # Set this flag to 'n' to disable the use of 'checkpaths'.  The default,
194 # if the 'N' option is not specified, is to run this test.
195 #CHECK_PATHS=y

197 # Set this flag to 'y' to enable the use of elfsigncmp to validate the
198 # output of elfsign.  Doing so requires that 't' be set in NIGHTLY_OPTIONS.
199 # The default is to not verify them.
200 #VERIFY_ELFSIGN=n

202 # BRINGOVER_FILES is the list of files nightly passes to bringover.
203 # If not set the default is "usr", but it can be used for bringing
204 # over deleted_files or other nifty directories.
205 #BRINGOVER_FILES="usr deleted_files"

207 # POST_NIGHTLY can be any command to be run at the end of nightly.  See
208 # nightly(1) for interactions between environment variables and this command.
209 #POST_NIGHTLY=
```

```
**********************************************************
    1475 Wed Oct 16 17:41:06 2013
new/usr/src/tools/findunref/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 2009 Sun Microsystems, Inc.  All rights reserved.
  24 # Use is subject to license terms.
  25 #

  27 PROG      = findunref
  28 MAN1FILES = findunref.1
  29 CFLAGS += $(CCVERBOSE)
  30 LINTFLAGS += -ux

  32 include ../Makefile.tools

  34 CERRWARN += -_gcc=-Wno-unused
  35 CERRWARN += -_gcc=-Wno-parentheses

  37 $(ROOTONBLDMAN1FILES) := FILEMODE=      644

  39 EXCEPTION_SRC= common open
  40 $(CLOSED_BUILD)EXCEPTION_SRC += closed
  40 EXCEPTION_LISTS=        $(EXCEPTION_SRC:%=exception_list.%)

  42 CLOBBERFILES += exception_list

  44 .KEEP_STATE:

  46 all:    $(PROG) exception_list

  48 install: all .WAIT $(ROOTONBLDMACHPROG) $(ROOTONBLDMAN1FILES)

  50 lint:   lint_PROG

  52 exception_list: $(EXCEPTION_LISTS)
  53         -$(RM) $@
  54         $(CAT) $(EXCEPTION_LISTS) > $@

  56 clean:
```



```
  58 include ../Makefile.targ
```

```
**********************************************************
    2213 Wed Oct 16 17:41:06 2013
new/usr/src/tools/findunref/exception_list.closed
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
#

#
# closed-tree exception list
#
# See README.exception_lists for details
#

#
# Ignore files that get used during a EXPORT_SRC or CRYPT_SRC build only.
#
./usr/closed/uts/sun4v/io/n2cp/Makefile
./usr/closed/uts/sun4v/io/ncp/Makefile

#
# Ignore files that are only used for warlock
#
./usr/closed/uts/sparc/marvell88sx/Makefile

#
# An unfortunate artifact of the bridged, split gate is that closed-source
# deleted files go where findunref can accidentally find them...
#
# And an unfortunate artifact of using these tools with both full Teamware
# and split, non-Teamware workspaces is that sometimes closed/deleted_files
# won't exist, so we need the ISUSED directive here.
#
# ISUSED - let checkpaths know that the next entry is good.
./usr/closed/deleted_files

#
# Ignore some files originally shared by Broadcom as part of bnxe driver
#
./usr/closed/uts/common/io/bnxe/577xx
./usr/closed/uts/common/io/bnxe/src/bnxe_debug.h
```



```
#
# Ignore some files originally shared by Broadcom as part of bnx driver
#
./usr/closed/uts/common/io/bnx/hsi.h
./usr/closed/uts/common/io/bnx/invm_cfg.h
./usr/closed/uts/common/io/bnx/iparms.h
./usr/closed/uts/common/io/bnx/itypes.h
./usr/closed/uts/common/io/bnx/status_code.h
./usr/closed/uts/common/io/bnx/tcp_ctx.h
./usr/closed/uts/common/io/bnx/toe_ctx.h
./usr/closed/uts/common/io/bnx/bnxdbg.c

#
# Ignore Makefile.tonic - not used unless we're running an
# OpenSolaris closedbins delivery build
#
./usr/closed/Makefile.tonic
```

```
**********************************************************
    5024 Wed Oct 16 17:41:07 2013
new/usr/src/tools/install.bin/install.bin.c
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
 */


#include <stdio.h>
#include <stdlib.h>
#include <strings.h>
#include <sys/param.h>
#include <fcntl.h>
#include <sys/errno.h>
#include <sys/types.h>
#include <sys/uio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <errno.h>
#include <libgen.h>
#include "stdusers.h"


#define FILE_BUFF       40960

static int suppress = 0;

static void usage(void);
static void file_copy(char *src_file, char *dest_file);
static void chown_file(const char *file, const char *group, const char *owner);
static void formclosed(char *root, char *closedroot);
static char *find_basename(const char *str);
static int creatdir(char *fn);


void
usage(void)
{
        (void) fprintf(stderr,
            "usage: install [-sd][-m mode][-g group][-u owner] "
            "usage: install [-sdO][-m mode][-g group][-u owner] "
            "-f dir file ...\n");
}
```



```
_____unchanged_portion_omitted_


void
formclosed(char *root, char *closedroot)
{
        int wholelen, residlen;
        char *temp;

        wholelen = strlen(root);
        temp = strstr(strstr(root, "proto/root_"), "/");
        temp++;
        temp = strstr(temp, "/");
        residlen = strlen(temp);
        (void) strlcpy(closedroot, root, wholelen - residlen + 1);
        (void) strlcat(closedroot, "-closed", MAXPATHLEN);
        (void) strlcat(closedroot, temp, MAXPATHLEN);
}


char *
find_basename(const char *str)
{
        int     i;
        int     len;

        len = strlen(str);

        for (i = len-1; i >= 0; i--)
                if (str[i] == '/')
                        return ((char *)(str + i + 1));
        return ((char *)str);
}
_____unchanged_portion_omitted_


int
main(int argc, char **argv)
{
        int     c;
        int     errflg = 0;
        int     dirflg = 0;
        char    *group = NULL;
        char    *owner = NULL;
        char    *dirb = NULL;
        char    *ins_file = NULL;
        int     mode = -1;
        char    dest_file[MAXPATHLEN];
        char    shadow_dest[MAXPATHLEN];
        char    shadow_dirb[MAXPATHLEN];
        int     tonic = 0;
        int     rv = 0;

        while ((c = getopt(argc, argv, "f:sm:du:g:")) != EOF) {
        while ((c = getopt(argc, argv, "f:sm:du:g:O")) != EOF) {
                switch (c) {
                case 'f':
                        dirb = optarg;
                        break;
                case 'g':
                        group = optarg;
                        break;
                case 'u':
                        owner = optarg;
```

```
180                         break;
181                 case 'd':
182                         dirflg = 1;
183                         break;
184                 case 'm':
185                         mode = strtol(optarg, NULL, 8);
186                         break;
187                 case 's':
188                         suppress = 1;
189                         break;
211                 case 'O':
212                         tonic = 1;
213                         break;
190                 case '?':
191                         errflg++;
192                         break;
193                 }
194         }

196         if (errflg) {
197                 usage();
198                 return (1);
199         }

201         if (argc == optind) {
202                 usage();
203                 return (1);
204         }

206         if (!dirflg && (dirb == NULL)) {
207                 (void) fprintf(stderr,
208                     "install: no destination directory specified.\n");
209                 return (1);
210         }

212         for (c = optind; c < argc; c++) {
213                 ins_file = argv[c];

215                 if (dirflg) {
240                         if (tonic) {
241                                 formclosed(ins_file, shadow_dest);
242                                 rv = creatdir(shadow_dest);
243                                 if (rv) {
244                                         (void) fprintf(stderr,
245                                             "install: tonic creatdir "
246                                             "%s (%d): (%s)\n",
247                                             shadow_dest, errno,
248                                             strerror(errno));
249                                         return (rv);
250                                 }
251                         }
216                         rv = creatdir(ins_file);
217                         if (rv) {
218                                 (void) fprintf(stderr,
219                                     "install: creatdir %s (%d): %s\n",
220                                     ins_file, errno, strerror(errno));
221                                 return (rv);
222                         }
223                         (void) strlcpy(dest_file, ins_file, MAXPATHLEN);

225                 } else {
226                         (void) strcat(strcat(strcpy(dest_file, dirb), "/"),
227                             find_basename(ins_file));
228                         file_copy(ins_file, dest_file);

266                         if (tonic) {
```



```
267                                 formclosed(dirb, shadow_dirb);
268                                 /*
269                                  * The standard directories in the proto
270                                  * area are created as part of "make setup",
271                                  * but that doesn't create them in the
272                                  * closed proto area. So if the target
273                                  * directory doesn't exist, we need to
274                                  * create it now.
275                                  */
276                                 rv = creatdir(shadow_dirb);
277                                 if (rv) {
278                                         (void) fprintf(stderr,
279                                             "install: tonic creatdir(f) "
280                                             "%s (%d): %s\n",
281                                             shadow_dirb, errno,
282                                             strerror(errno));
283                                         return (rv);
284                                 }
285                                 (void) strcat(strcat(strcpy(shadow_dest,
286                                     shadow_dirb), "/"),
287                                     find_basename(ins_file));
288                                 file_copy(ins_file, shadow_dest);
289                         }
229                 }

231                 if (group || owner)
292                 if (group || owner) {
232                         chown_file(dest_file, group, owner);

294                         if (tonic)
295                                 chown_file(shadow_dest, group, owner);
296                 }
234                 if (mode != -1) {
235                         (void) umask(0);
236                         if (chmod(dest_file, mode) == -1) {
237                                 (void) fprintf(stderr,
238                                     "install: chmod of %s to mode %o failed "
239                                     "(%d): %s\n",
240                                     dest_file, mode, errno, strerror(errno));
241                                 return (1);
242                         }
306                         if (tonic) {
307                                 if (chmod(shadow_dest, mode) == -1) {
308                                         (void) fprintf(stderr,
309                                             "install: tonic chmod of %s "
310                                             "to mode %o failed (%d): %s\n",
311                                             shadow_dest, mode,
312                                             errno, strerror(errno));
313                                         return (1);
314                                 }
315                         }
243                 }
244         }
245         return (0);
246 }
```

_____*unchanged_portion_omitted_*

```
**********************************************************
   25470 Wed Oct 16 17:41:07 2013
new/usr/src/tools/scripts/Install.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
______unchanged_portion_omitted_

690 function copy_kmdb {
691         typeset kmdbtgtdir=$INSTALL_FILES/platform/$KARCH/$GLOMNAME/misc
692         typeset bitdirs=
693         typeset isadir=
694         typeset b64srcdir=
695         typeset b64tgtdir=
696         typeset b32srcdir=
697         typeset b32tgtdir=
698         typeset machdir=
699         typeset platdir=

701         if [[ $KMDB = "no" || ! -d $SRC/cmd/mdb ]] ; then
702                 # The kmdb copy was suppressed or the workspace doesn't contain
703                 # the mdb subtree.  Either way, there's nothing to do.
704                 STATE=2
705                 save_state
706                 return
707         fi

709         if [[ $(mach) = "i386" ]] ; then
710                 isadir="intel"
711                 b64srcdir="amd64"
712                 b64tgtdir="amd64"
713                 b32srcdir="ia32"
714                 b32tgtdir="."
715         else
716                 isadir="sparc"
717                 b64srcdir="v9"
718                 b64tgtdir="sparcv9"
719                 b32srcdir="v7"
720                 b32tgtdir="."
721         fi

723         typeset foundkmdb=no
724         typeset kmdbpath=
725         typeset destdir=

727         platdir=$INSTALL_FILES/platform/$KARCH/$GLOMNAME
728         if [[ $GLOM = "yes" ]] ; then
729                 machdir=$platdir
730         else
731                 machdir=$INSTALL_FILES/kernel
732         fi

734         srctrees=$SRC
735         if [[ -d $SRC/../closed && "$CLOSED_IS_PRESENT" != no ]]; then
736                 srctrees="$srctrees $SRC/../closed"
737         else
735         if [ -z "$ON_CRYPTO_BINS" ]; then
736                 echo "Warning: ON_CRYPTO_BINS not set; pre-signed" \
737                     "crypto not provided."
738         fi
742         fi
739         if [[ $WANT64 = "yes" ]] ; then
740                 # kmdbmod for sparc and x86 are built and installed
741                 # in different places
```



```
742                 if [[ $(mach) = "i386" ]] ; then
743                         kmdbpath=$SRC/cmd/mdb/$isadir/$b64srcdir/kmdb/kmdbmod
744                         destdir=$machdir/misc/$b64tgtdir
745                 else
746                         kmdbpath=$SRC/cmd/mdb/$KARCH/$b64srcdir/kmdb/kmdbmod
747                         destdir=$platdir/misc/$b64tgtdir
748                 fi

750                 if kmdb_copy_kmdbmod $kmdbpath $destdir ; then
751                         foundkmdb="yes"

753                         for tree in $srctrees; do
754                                 kmdb_copy_machkmods \
755                                     $tree/cmd/mdb/$isadir/$b64srcdir \
756                                     $machdir/kmdb/$b64tgtdir
757                                 kmdb_copy_karchkmods $tree/cmd/mdb/$KARCH \
758                                     $platdir/kmdb/$b64tgtdir $b64srcdir
759                         done
760                 fi
761         fi

763         if [[ $WANT32 = "yes" ]] ; then
764                 kmdbpath=$SRC/cmd/mdb/$isadir/$b32srcdir/kmdb/kmdbmod
765                 destdir=$machdir/misc/$b32tgtdir

767                 if kmdb_copy_kmdbmod $kmdbpath $destdir ; then
768                         foundkmdb="yes"

770                         for tree in $srctrees; do
771                                 kmdb_copy_machkmods \
772                                     $tree/cmd/mdb/$isadir/$b32srcdir \
773                                     $machdir/kmdb/$b32tgtdir
774                                 kmdb_copy_karchkmods $tree/cmd/mdb/$KARCH \
775                                     $platdir/kmdb/$b32tgtdir $b32srcdir
776                         done
777                 fi
778         fi

780         # A kmdb-less workspace isn't fatal, but it is potentially problematic,
781         # as the changes made to uts may have altered something upon which kmdb
782         # depends.  We will therefore remind the user that they haven't built it
783         # yet.
784         if [[ $foundkmdb != "yes" ]] ; then
785                 echo "WARNING: kmdb isn't built, and won't be included"
786         fi

788         STATE=2
789         save_state
790         return
791 }
______unchanged_portion_omitted_
```

```
**********************************************************
   12468 Wed Oct 16 17:41:07 2013
new/usr/src/tools/scripts/bldenv.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
______unchanged_portion_omitted_
 154 [+SEE ALSO?\bnightly\b(1)]
 155 '

 157 # main
 158 builtin basename

 160 # boolean flags (true/false)
 161 typeset flags=(
 162         typeset c=false
 163         typeset f=false
 164         typeset d=false
 165         typeset O=false
 166         typeset o=false
 167         typeset t=true
 168         typeset s=(
 169                 typeset e=false
 170                 typeset h=false
 171                 typeset d=false
 172                 typeset o=false
 173         )
 174 )

 176 typeset progname="$(basename -- "${0}")"

 178 OPTIND=1
 179 SUFFIX="-nd"

 181 while getopts -a "${progname}" "${USAGE}" OPT ; do
 182     case ${OPT} in
 183           c)    flags.c=true  ;;
 184           +c)   flags.c=false ;;
 185           f)    flags.f=true  ;;
 186           +f)   flags.f=false ;;
 187           d)    flags.d=true  SUFFIX=""    ;;
 188           +d)   flags.d=false SUFFIX="-nd" ;;
 189           t)    flags.t=true  ;;
 190           +t)   flags.t=false ;;
 191           S)    set_S_flag "$OPTARG" ;;
 192           \?)   usage ;;
 193     esac
 194 done
 195 shift $((OPTIND-1))

 197 # test that the path to the environment-setting file was given
 198 if (( $# < 1 )) ; then
 199         usage
 200 fi

 202 # force locale to C
 203 export \
 204         LC_COLLATE=C \
 205         LC_CTYPE=C \
 206         LC_MESSAGES=C \
 207         LC_MONETARY=C \
 208         LC_NUMERIC=C \
 209         LC_TIME=C
```



```
 211 # clear environment variables we know to be bad for the build
 212 unset \
 213         LD_OPTIONS \
 214         LD_LIBRARY_PATH \
 215         LD_AUDIT \
 216         LD_BIND_NOW \
 217         LD_BREADTH \
 218         LD_CONFIG \
 219         LD_DEBUG \
 220         LD_FLAGS \
 221         LD_LIBRARY_PATH_64 \
 222         LD_NOVERSION \
 223         LD_ORIGIN \
 224         LD_LOADFLTR \
 225         LD_NOAUXFLTR \
 226         LD_NOCONFIG \
 227         LD_NODIRCONFIG \
 228         LD_NOOBJALTER \
 229         LD_PRELOAD \
 230         LD_PROFILE \
 231         CONFIG \
 232         GROUP \
 233         OWNER \
 234         REMOTE \
 235         ENV \
 236         ARCH \
 237         CLASSPATH

 239 #
 240 # Setup environment variables
 241 #
 242 if [[ -f /etc/nightly.conf ]]; then
 243         source /etc/nightly.conf
 244 fi

 246 if [[ -f "$1" ]]; then
 247         if [[ "$1" == */* ]]; then
 248                 source "$1"
 249         else
 250                 source "./$1"
 251         fi
 252 else
 253         if [[ -f "/opt/onbld/env/$1" ]]; then
 254                 source "/opt/onbld/env/$1"
 255         else
 256                 printf \
 257                     'Cannot find env file as either %s or /opt/onbld/env/%s\n' \
 258                     "$1" "$1"
 259                 exit 1
 260         fi
 261 fi
 262 shift

 264 # contents of stdenv.sh inserted after next line:
 265 # STDENV_START
 266 # STDENV_END

 268 # Check if we have sufficient data to continue...
 269 [[ -v CODEMGR_WS ]] || fatal_error "Error: Variable CODEMGR_WS not set."
 270 [[ -d "${CODEMGR_WS}" ]] || fatal_error "Error: ${CODEMGR_WS} is not a directory
 271 [[ -f "${CODEMGR_WS}/usr/src/Makefile" ]] || fatal_error "Error: ${CODEMGR_WS}/u

 273 # must match the getopts in nightly.sh
 274 OPTIND=1
 275 NIGHTLY_OPTIONS="-${NIGHTLY_OPTIONS#-}"
 276 while getopts '+0AaBCDdFfGIilMmNnOopRrS:tUuWwXxz' FLAG "$NIGHTLY_OPTIONS"
```

```
277 do
278         case "$FLAG" in
279           O)    flags.O=true  ;;
280           +O)   flags.O=false ;;
281           o)    flags.o=true  ;;
282           +o)   flags.o=false ;;
283           t)    flags.t=true  ;;
284           +t)   flags.t=false ;;
285           S)    set_S_flag "$OPTARG" ;;
286           *)    ;;
287         esac
288 done

290 POUND_SIGN="#"
291 # have we set RELEASE_DATE in our env file?
292 if [ -z "$RELEASE_DATE" ]; then
293         RELEASE_DATE=$(LC_ALL=C date +"%B %Y")
294 fi
295 BUILD_DATE=$(LC_ALL=C date +%Y-%b-%d)
296 BASEWSDIR=$(basename -- "${CODEMGR_WS}")
297 DEV_CM="\"@(#)SunOS Internal Development: $LOGNAME $BUILD_DATE [$BASEWSDIR]\""
298 export DEV_CM RELEASE_DATE POUND_SIGN

300 export INTERNAL_RELEASE_BUILD=

302 print 'Build type   is  \c'
303 if ${flags.d} ; then
304         print 'DEBUG'
305         unset RELEASE_BUILD
306         unset EXTRA_OPTIONS
307         unset EXTRA_CFLAGS
308 else
309         # default is a non-DEBUG build
310         print 'non-DEBUG'
311         export RELEASE_BUILD=
312         unset EXTRA_OPTIONS
313         unset EXTRA_CFLAGS
314 fi

316 [[ "${flags.O}" == "true" ]] && export MULTI_PROTO="yes"

318 # update build-type variables
319 PKGARCHIVE="${PKGARCHIVE}${SUFFIX}"

321 # Append source version
322 if "${flags.s.e}" ; then
323         VERSION+=":EXPORT"
324         SRC="${EXPORT_SRC}/usr/src"
325 fi
326
327 if "${flags.s.d}" ; then
328         VERSION+=":DOMESTIC"
329         SRC="${EXPORT_SRC}/usr/src"
330 fi

332 if "${flags.s.h}" ; then
333         VERSION+=":HYBRID"
334         SRC="${EXPORT_SRC}/usr/src"
335 fi
336
337 if "${flags.s.o}" ; then
338         VERSION+=":OPEN_ONLY"
339         SRC="${OPEN_SRCDIR}/usr/src"
340 fi

342 #       Set PATH for a build
```



```
343 PATH="/opt/onbld/bin:/opt/onbld/bin/${MACH}:/opt/SUNWspro/bin:/usr/ccs/bin:/usr/
344 if [[ "${SUNWSPRO}" != "" ]]; then
345         export PATH="${SUNWSPRO}/bin:$PATH"
346 fi

348 if [[ -z "$CLOSED_IS_PRESENT" ]]; then
349         if [[ -d $SRC/../closed ]]; then
350                 export CLOSED_IS_PRESENT="yes"
351         else
352                 export CLOSED_IS_PRESENT="no"
353         fi
354 fi

348 TOOLS="${SRC}/tools"
349 TOOLS_PROTO="${TOOLS}/proto/root_${MACH}-nd" ; export TOOLS_PROTO

351 if "${flags.t}" ; then
352         export ONBLD_TOOLS="${ONBLD_TOOLS:=${TOOLS_PROTO}/opt/onbld}"

354         export STABS="${TOOLS_PROTO}/opt/onbld/bin/${MACH}/stabs"
355         export CTFSTABS="${TOOLS_PROTO}/opt/onbld/bin/${MACH}/ctfstabs"
356         export GENOFFSETS="${TOOLS_PROTO}/opt/onbld/bin/genoffsets"

358         export CTFCONVERT="${TOOLS_PROTO}/opt/onbld/bin/${MACH}/ctfconvert"
359         export CTFMERGE="${TOOLS_PROTO}/opt/onbld/bin/${MACH}/ctfmerge"

361         export CTFCVTPTBL="${TOOLS_PROTO}/opt/onbld/bin/ctfcvtptbl"
362         export CTFFINDMOD="${TOOLS_PROTO}/opt/onbld/bin/ctffindmod"

364         PATH="${TOOLS_PROTO}/opt/onbld/bin/${MACH}:${PATH}"
365         PATH="${TOOLS_PROTO}/opt/onbld/bin:${PATH}"
366         export PATH
367 fi

369 export DMAKE_MODE=${DMAKE_MODE:-parallel}

371 if "${flags.o}" ; then
372         export CH=
373 else
374         unset CH
375 fi
376 DEF_STRIPFLAG="-s"

378 TMPDIR="/tmp"

380 # "o_FLAG" is used by "nightly.sh" (it may be useful to rename this
381 # variable using a more descriptive name later)
382 export o_FLAG="$(${flags.o} && print 'y' || print 'n')"

384 export \
385         PATH TMPDIR \
386         POUND_SIGN \
387         DEF_STRIPFLAG \
388         RELEASE_DATE
389 unset \
390         CFLAGS \
391         LD_LIBRARY_PATH

393 # a la ws
394 ENVLDLIBS1=
395 ENVLDLIBS2=
396 ENVLDLIBS3=
397 ENVCPPFLAGS1=
398 ENVCPPFLAGS2=
399 ENVCPPFLAGS3=
400 ENVCPPFLAGS4=
```

```
401 PARENT_ROOT=
402 PARENT_TOOLS_ROOT=

404 if [[ "$MULTI_PROTO" != "yes" && "$MULTI_PROTO" != "no" ]]; then
405         printf \
406             'WARNING: invalid value for MULTI_PROTO (%s); setting to "no".\n' \
407             "$MULTI_PROTO"
408         export MULTI_PROTO="no"
409 fi

411 [[ "$MULTI_PROTO" == "yes" ]] && export ROOT="${ROOT}${SUFFIX}"

421 export TONICBUILD="#"

413 if "${flags.O}" ; then
424         if [[ "$CLOSED_IS_PRESENT" != "yes" ]]; then
414         print "OpenSolaris closed binary generation requires "
415         print "closed tree"
416         exit 1
428         fi
429         print "Generating OpenSolaris deliverables"
430         # We only need CLOSEDROOT in the env for convenience. Makefile.master
431         # figures out what it needs when it matters.
432         export CLOSEDROOT="${ROOT}-closed"
433         export TONICBUILD=""
417 fi

419 ENVLDLIBS1="-L$ROOT/lib -L$ROOT/usr/lib"
420 ENVCPPFLAGS1="-I$ROOT/usr/include"
421 MAKEFLAGS=e

423 export \
424         ENVLDLIBS1 \
425         ENVLDLIBS2 \
426         ENVLDLIBS3 \
427         ENVCPPFLAGS1 \
428         ENVCPPFLAGS2 \
429         ENVCPPFLAGS3 \
430         ENVCPPFLAGS4 \
431         MAKEFLAGS \
432         PARENT_ROOT \
433         PARENT_TOOLS_ROOT

435 printf 'RELEASE      is %s\n'   "$RELEASE"
436 printf 'VERSION      is %s\n'   "$VERSION"
437 printf 'RELEASE_DATE is %s\n\n' "$RELEASE_DATE"

439 if [[ -f "$SRC/Makefile" ]] && egrep -s '^setup:' "$SRC/Makefile" ; then
440         print "The top-level 'setup' target is available \c"
441         print "to build headers and tools."
442         print ""

444 elif "${flags.t}" ; then
445         printf \
446             'The tools can be (re)built with the install target in %s.\n\n' \
447             "${TOOLS}"
448 fi

450 #
451 # place ourselves in a new task, respecting BUILD_PROJECT if set.
452 #
453 /usr/bin/newtask -c $$ ${BUILD_PROJECT:+-p$BUILD_PROJECT}

455 if [[ "${flags.c}" == "false" && -x "$SHELL" && \
456     "$(basename -- "${SHELL}")" != "csh" ]]; then
457         # $SHELL is set, and it's not csh.
```



```
459         if "${flags.f}" ; then
460                 print 'WARNING: -f is ignored when $SHELL is not csh'
461         fi

463         printf 'Using %s as shell.\n' "$SHELL"
464         exec "$SHELL" ${@:+-c "$@"}

466 elif "${flags.f}" ; then
467         print 'Using csh -f as shell.'
468         exec csh -f ${@:+-c "$@"}

470 else
471         print 'Using csh as shell.'
472         exec csh ${@:+-c "$@"}
473 fi

475 # not reached
```

```
**********************************************************
    3889 Wed Oct 16 17:41:07 2013
new/usr/src/tools/scripts/checkpaths.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#

# Quis custodiet ipsos custodies?

if [ -z "$SRC" ]; then
        SRC=$CODEMGR_WS/usr/src
fi

if [ -z "$CODEMGR_WS" -o ! -d "$CODEMGR_WS" -o ! -d "$SRC" ]; then
        echo "$0: must be run from within a workspace."
        exit 1
fi

cd $CODEMGR_WS || exit 1

# Use -b to tell this script to ignore derived (built) objects.
if [ "$1" = "-b" ]; then
        b_flg=y
fi

# Not currently used; available for temporary workarounds.
args="-k NEVER_CHECK"

# We intentionally don't depend on $MACH here, and thus no $ROOT.  If
# a proto area exists, then we use it.  This allows this script to be
# run against gates (which should contain both SPARC and x86 proto
# areas), build workspaces (which should contain just one proto area),
# and unbuilt workspaces (which contain no proto areas).
if [ "$b_flg" = y ]; then
        rootlist=
elif [ $# -gt 0 ]; then
        rootlist=$*
else
```



```
        rootlist="$CODEMGR_WS/proto/root_sparc $CODEMGR_WS/proto/root_i386"
fi

# If the closed source is not present, then exclude IKE from validation.
if [ "$CLOSED_IS_PRESENT" = no ]; then
        excl="-e ^usr/include/ike/"
fi

for ROOT in $rootlist
do
        case "$ROOT" in
        *sparc|*sparc-nd)
                arch=sparc
                ;;
        *i386|*i386-nd)
                arch=i386
                ;;
        *)
                echo "$ROOT has unknown architecture." >&2
                exit 1
                ;;
        esac
        if [ -d $ROOT ]; then
                #
                # This is the old-style packaging exception list, from
                # the svr4-specific usr/src/pkgdefs
                #
                [ -f $SRC/pkgdefs/etc/exception_list_$arch ] && \
                        validate_paths '-s/\s*'$arch'$//'  \
                            -e ^usr/include/ike/ -b $ROOT \
                        validate_paths '-s/\s*'$arch'$//' $excl -b $ROOT \
                            $args $SRC/pkgdefs/etc/exception_list_$arch
                #
                # These are the new-style packaging exception lists,
                # from the repository-wide exception_lists/ directory.
                #
                e="$CODEMGR_WS/exception_lists/packaging"
                for f in $e; do
                        if [ -f $f ]; then
                                nawk 'NF == 1 || /[     ]\+'$arch'$/ { print; }'
                                    < $f | validate_paths -b $ROOT -n $f
                        fi
                done
        fi
done

# Two entries in the findunref exception_list deal with things created
# by nightly.  Otherwise, this test could be run on an unmodifed (and
# unbuilt) workspace.  We handle this by flagging the one that is
# present only on a built workspace (./*.out) and the one that's
# present only after a run of findunref (./*.ref) with ISUSED, and
# disabling all checks of them.  The assumption is that the entries
# marked with ISUSED are always known to be good, thus the Latin quote
# at the top of the file.
#
# The exception_list is generated from whichever input files are appropriate
# for this workspace, so checking it obviates the need to check the inputs.

if [ -r $SRC/tools/findunref/exception_list ]; then
        validate_paths -k ISUSED -r -e '^\*' $SRC/tools/findunref/exception_list
fi

if [ -f $SRC/tools/opensolaris/license-list ]; then
        excl=
        if [ "$CLOSED_IS_PRESENT" = no ]; then
                excl="-e ^usr/closed"
```

```
        fi
        sed -e 's/$/.descrip/' < $SRC/tools/opensolaris/license-list | \
                validate_paths -n SRC/tools/opensolaris/license-list \
                    -e ^usr/closed
                validate_paths -n SRC/tools/opensolaris/license-list $excl
fi

# Finally, make sure the that (req|inc).flg files are in good shape.
# If SCCS files are not expected to be present, though, then don't
# check them.
if [ ! -d "$CODEMGR_WS/Codemgr_wsdata" ]; then
        f_flg='-f'
fi
# If the closed source is not present, then don't validate it.
if [ "$CLOSED_IS_PRESENT" = no ]; then
        excl="-e ^usr/closed/"
fi

validate_flg $f_flg -e ^usr/closed/
validate_flg $f_flg $excl

exit 0
```

```
**********************************************************
   80188 Wed Oct 16 17:41:07 2013
new/usr/src/tools/scripts/nightly.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
______unchanged_portion_omitted_

 296 #
 297 # Mercurial-specific copy code for copy_source().
 297 # Mercurial-specific copy code for copy_source().  Handles the
 298 # combined open and closed trees.
 298 #
 299 # Returns 0 for success, non-zero for failure.
 300 #
 301 # usage: copy_source_mercurial destdir srcroot
 302 #
 303 function copy_source_mercurial {
 304         typeset dest=$1
 305         typeset srcroot=$2
 307         typeset open_top closed_top

 307         hg locate -I "$srcroot" | cpio -pd "$dest" >>$LOGFILE 2>&1
 309         case $srcroot in
 310         usr)
 311                 open_top=usr
 312                 if [[ "$CLOSED_IS_PRESENT" = yes ]]; then
 313                         closed_top=usr/closed
 314                 fi
 315                 ;;
 316         usr/closed*)
 317                 if [[ "$CLOSED_IS_PRESENT" = no ]]; then
 318                         printf "can't copy %s: closed tree not present.\n" \
 319                             "$srcroot" | tee -a $mail_msg_file >> $LOGFILE
 320                         return 1
 321                 fi
 322                 closed_top="$srcroot"
 323                 ;;
 324         *)
 325                 open_top="$srcroot"
 326                 ;;
 327         esac

 329         if [[ -n "$open_top" ]]; then
 330                 hg locate -I "$open_top" | cpio -pd "$dest" >>$LOGFILE 2>&1
 308         if (( $? != 0 )) ; then
 309             printf "cpio failed for %s\n" "$dest" |
 310                 tee -a $mail_msg_file >> $LOGFILE
 311             return 1
 312         fi
 336         fi

 338         if [[ -n "$closed_top" ]]; then
 339                 mkdir -p "$dest/usr/closed" || return 1
 340                 if [[ "$closed_top" = usr/closed ]]; then
 341                         (cd usr/closed; hg locate |
 342                             cpio -pd "$dest/usr/closed") >>$LOGFILE 2>&1
 343                         if (( $? != 0 )) ; then
 344                             printf "cpio failed for %s/usr/closed\n" \
 345                                 "$dest" | tee -a $mail_msg_file >> $LOGFILE
 346                             return 1
 347                         fi
 348                 else
 349                         # copy subtree of usr/closed
```



```
 350                         closed_top=${closed_top#usr/closed/}
 351                         (cd usr/closed; hg locate -I "$closed_top" |
 352                             cpio -pd "$dest/usr/closed") >>$LOGFILE 2>&1
 353                         if (( $? != 0 )) ; then
 354                             printf "cpio failed for %s/usr/closed/%s\n" \
 355                                 "$dest" "$closed_top" |
 356                                 tee -a $mail_msg_file >> $LOGFILE
 357                             return 1
 358                         fi
 359                 fi
 360         fi

 314         return 0
 315 }
______unchanged_portion_omitted_

 846 #
 847 # Verify that the closed tree is present if it needs to be.
 896 # Sets CLOSED_IS_PRESENT for future use.
 848 #
 849 function check_closed_tree {
 850         if [[ ! -d "$ON_CLOSED_BINS" ]]; then
 851                 echo "ON_CLOSED_BINS must point to the closed binaries tree."
 899         if [ -z "$CLOSED_IS_PRESENT" ]; then
 900                 if [ -d $CODEMGR_WS/usr/closed ]; then
 901                         CLOSED_IS_PRESENT="yes"
 902                 else
 903                         CLOSED_IS_PRESENT="no"
 904                 fi
 905                 export CLOSED_IS_PRESENT
 906         fi
 907         if [[ "$CLOSED_IS_PRESENT" = no && ! -d "$ON_CLOSED_BINS" ]]; then
 908                 #
 909                 # If it's an old (pre-split) tree or an empty
 910                 # workspace, don't complain.
 911                 #
 912                 if grep -s CLOSED_BUILD $SRC/Makefile.master > /dev/null; then
 913                         echo "If the closed sources are not present," \
 914                             "ON_CLOSED_BINS"
 915                         echo "must point to the closed binaries tree."
 852                 build_ok=n
 853                 exit 1
 854         fi
 919         fi
 855 }
______unchanged_portion_omitted_

1041 OPTIND=1
1042 while getopts +inS:tV: FLAG
1043 do
1044         case $FLAG in
1045           i )   i_FLAG=y; i_CMD_LINE_FLAG=y
1046                 ;;
1047           n )   n_FLAG=y
1048                 ;;
1049           S )
1050                 set_S_flag $OPTARG
1051                 ;;
1052          +t )   t_FLAG=n
1053                 ;;
1054           V )   V_FLAG=y
1055                 V_ARG="$OPTARG"
1056                 ;;
1057          \? )   echo "$USAGE"
1058                 exit 1
1059                 ;;
```

```
1060         esac
1061 done

1063 # correct argument count after options
1064 shift `expr $OPTIND - 1`

1066 # test that the path to the environment-setting file was given
1067 if [ $# -ne 1 ]; then
1068         echo "$USAGE"
1069         exit 1
1070 fi

1072 # check if user is running nightly as root
1073 # ISUSER is set non-zero if an ordinary user runs nightly, or is zero
1074 # when root invokes nightly.
1075 /usr/bin/id | grep '^uid=0(' >/dev/null 2>&1
1076 ISUSER=$?;      export ISUSER

1078 #
1079 # force locale to C
1080 LC_COLLATE=C;   export LC_COLLATE
1081 LC_CTYPE=C;     export LC_CTYPE
1082 LC_MESSAGES=C;  export LC_MESSAGES
1083 LC_MONETARY=C;  export LC_MONETARY
1084 LC_NUMERIC=C;   export LC_NUMERIC
1085 LC_TIME=C;      export LC_TIME

1087 # clear environment variables we know to be bad for the build
1088 unset LD_OPTIONS
1089 unset LD_AUDIT          LD_AUDIT_32             LD_AUDIT_64
1090 unset LD_BIND_NOW       LD_BIND_NOW_32          LD_BIND_NOW_64
1091 unset LD_BREADTH        LD_BREADTH_32           LD_BREADTH_64
1092 unset LD_CONFIG         LD_CONFIG_32            LD_CONFIG_64
1093 unset LD_DEBUG          LD_DEBUG_32             LD_DEBUG_64
1094 unset LD_DEMANGLE       LD_DEMANGLE_32          LD_DEMANGLE_64
1095 unset LD_FLAGS          LD_FLAGS_32             LD_FLAGS_64
1096 unset LD_LIBRARY_PATH   LD_LIBRARY_PATH_32      LD_LIBRARY_PATH_64
1097 unset LD_LOADFLTR       LD_LOADFLTR_32          LD_LOADFLTR_64
1098 unset LD_NOAUDIT        LD_NOAUDIT_32           LD_NOAUDIT_64
1099 unset LD_NOAUXFLTR      LD_NOAUXFLTR_32         LD_NOAUXFLTR_64
1100 unset LD_NOCONFIG       LD_NOCONFIG_32          LD_NOCONFIG_64
1101 unset LD_NODIRCONFIG    LD_NODIRCONFIG_32       LD_NODIRCONFIG_64
1102 unset LD_NODIRECT       LD_NODIRECT_32          LD_NODIRECT_64
1103 unset LD_NOLAZYLOAD     LD_NOLAZYLOAD_32        LD_NOLAZYLOAD_64
1104 unset LD_NOOBJALTER     LD_NOOBJALTER_32        LD_NOOBJALTER_64
1105 unset LD_NOVERSION      LD_NOVERSION_32         LD_NOVERSION_64
1106 unset LD_ORIGIN         LD_ORIGIN_32            LD_ORIGIN_64
1107 unset LD_PRELOAD        LD_PRELOAD_32           LD_PRELOAD_64
1108 unset LD_PROFILE        LD_PROFILE_32           LD_PROFILE_64

1110 unset CONFIG
1111 unset GROUP
1112 unset OWNER
1113 unset REMOTE
1114 unset ENV
1115 unset ARCH
1116 unset CLASSPATH
1117 unset NAME

1119 #
1120 # To get ONBLD_TOOLS from the environment, it must come from the env file.
1121 # If it comes interactively, it is generally TOOLS_PROTO, which will be
1122 # clobbered before the compiler version checks, which will therefore fail.
1123 #
1124 unset ONBLD_TOOLS
```



```
1126 #
1127 #       Setup environmental variables
1128 #
1129 if [ -f /etc/nightly.conf ]; then
1130         . /etc/nightly.conf
1131 fi

1133 if [ -f $1 ]; then
1134         if [[ $1 = */* ]]; then
1135                 . $1
1136         else
1137                 . ./$1
1138         fi
1139 else
1140         if [ -f $OPTHOME/onbld/env/$1 ]; then
1141                 . $OPTHOME/onbld/env/$1
1142         else
1143                 echo "Cannot find env file as either $1 or $OPTHOME/onbld/env/$1
1144                 exit 1
1145         fi
1146 fi

1148 # contents of stdenv.sh inserted after next line:
1149 # STDENV_START
1150 # STDENV_END

1152 # Check if we have sufficient data to continue...
1153 [[ -v CODEMGR_WS ]] || fatal_error "Error: Variable CODEMGR_WS not set."
1154 if  [[ "${NIGHTLY_OPTIONS}" == ~(F)n ]] ; then
1155         # Check if the gate data are valid if we don't do a "bringover" below
1156         [[ -d "${CODEMGR_WS}" ]] || \
1157                 fatal_error "Error: ${CODEMGR_WS} is not a directory."
1158         [[ -f "${CODEMGR_WS}/usr/src/Makefile" ]] || \
1159                 fatal_error "Error: ${CODEMGR_WS}/usr/src/Makefile not found."
1160 fi

1162 #
1163 # place ourselves in a new task, respecting BUILD_PROJECT if set.
1164 #
1165 if [ -z "$BUILD_PROJECT" ]; then
1166         /usr/bin/newtask -c $$
1167 else
1168         /usr/bin/newtask -c $$ -p $BUILD_PROJECT
1169 fi

1171 ps -o taskid= -p $$ | read build_taskid
1172 ps -o project= -p $$ | read build_project

1174 #
1175 # See if NIGHTLY_OPTIONS is set
1176 #
1177 if [ "$NIGHTLY_OPTIONS" = "" ]; then
1178         NIGHTLY_OPTIONS="-aBm"
1179 fi

1181 #
1182 # If BRINGOVER_WS was not specified, let it default to CLONE_WS
1183 #
1184 if [ "$BRINGOVER_WS" = "" ]; then
1185         BRINGOVER_WS=$CLONE_WS
1186 fi

1188 #
1189 # If CLOSED_BRINGOVER_WS was not specified, let it default to CLOSED_CLONE_WS
1190 #
1191 if [ "$CLOSED_BRINGOVER_WS" = "" ]; then
```

```
1192         CLOSED_BRINGOVER_WS=$CLOSED_CLONE_WS
1193 fi

1195 #
1196 # If BRINGOVER_FILES was not specified, default to usr
1197 #
1198 if [ "$BRINGOVER_FILES" = "" ]; then
1199         BRINGOVER_FILES="usr"
1200 fi

1267 #
1268 # If the closed sources are not present, the closed binaries must be
1269 # present for the build to succeed.  If there's no pointer to the
1270 # closed binaries, flag that now, rather than forcing the user to wait
1271 # a couple hours (or more) to find out.
1272 #
1273 orig_closed_is_present="$CLOSED_IS_PRESENT"
1202 check_closed_tree

1204 #
1205 # Note: changes to the option letters here should also be applied to the
1206 #       bldenv script.  'd' is listed for backward compatibility.
1207 #
1208 NIGHTLY_OPTIONS=-${NIGHTLY_OPTIONS#-}
1209 OPTIND=1
1210 while getopts +ABCDdFfGIilMmNnOoPpRrS:TtUuWwXxz FLAG $NIGHTLY_OPTIONS
1211 do
1212         case $FLAG in
1213           A )   A_FLAG=y
1286                 #
1287                 # If ELF_DATA_BASELINE_DIR is not defined, and we are on SWAN
1288                 # (based on CLOSED_IS_PRESENT), then refuse to run. The value
1289                 # of ELF version checking is greatly enhanced by including
1290                 # the baseline gate comparison.
1291                 if [ "$CLOSED_IS_PRESENT" = 'yes' -a \
1292                      "$ELF_DATA_BASELINE_DIR" = '' ]; then
1293                         echo "ELF_DATA_BASELINE_DIR must be set if the A" \
1294                             "flag is present in\nNIGHTLY_OPTIONS and closed" \
1295                             "sources are present. Update environment file."
1296                         exit 1;
1297                 fi
1214                 ;;
1215           B )   D_FLAG=y
1216                 ;; # old version of D
1217           C )   C_FLAG=y
1218                 ;;
1219           D )   D_FLAG=y
1220                 ;;
1221           F )   F_FLAG=y
1222                 ;;
1223           f )   f_FLAG=y
1224                 ;;
1225           G )   u_FLAG=y
1226                 ;;
1227           I )   m_FLAG=y
1228                 p_FLAG=y
1229                 u_FLAG=y
1230                 ;;
1231           i )   i_FLAG=y
1232                 ;;
1233           l )   l_FLAG=y
1234                 ;;
1235           M )   M_FLAG=y
1236                 ;;
1237           m )   m_FLAG=y
1238                 ;;
```



```
1239           N )   N_FLAG=y
1240                 ;;
1241           n )   n_FLAG=y
1242                 ;;
1243           O )   O_FLAG=y
1244                 ;;
1245           o )   o_FLAG=y
1246                 ;;
1247           P )   P_FLAG=y
1248                 ;; # obsolete
1249           p )   p_FLAG=y
1250                 ;;
1251           R )   m_FLAG=y
1252                 p_FLAG=y
1253                 ;;
1254           r )   r_FLAG=y
1255                 ;;
1256           S )
1257                 set_S_flag $OPTARG
1258                 ;;
1259           T )   T_FLAG=y
1260                 ;; # obsolete
1261          +t )   t_FLAG=n
1262                 ;;
1263           U )   if [ -z "${PARENT_ROOT}" ]; then
1264                         echo "PARENT_ROOT must be set if the U flag is" \
1265                             "present in NIGHTLY_OPTIONS."
1266                         exit 1
1267                 fi
1268                 NIGHTLY_PARENT_ROOT=$PARENT_ROOT
1269                 if [ -n "${PARENT_TOOLS_ROOT}" ]; then
1270                         NIGHTLY_PARENT_TOOLS_ROOT=$PARENT_TOOLS_ROOT
1271                 fi
1272                 U_FLAG=y
1273                 ;;
1274           u )   u_FLAG=y
1275                 ;;
1276           W )   W_FLAG=y
1277                 ;;

1279           w )   w_FLAG=y
1280                 ;;
1281           X )   # now that we no longer need realmode builds, just
1282                 # copy IHV packages.  only meaningful on x86.
1283                 if [ "$MACH" = "i386" ]; then
1284                         X_FLAG=y
1285                 fi
1286                 ;;
1287           x )   XMOD_OPT="-x"
1288                 ;;
1289          \? )   echo "$USAGE"
1290                 exit 1
1291                 ;;
1292         esac
1293 done

1295 if [ $ISUSER -ne 0 ]; then
1296         if [ "$o_FLAG" = "y" ]; then
1297                 echo "Old-style build requires root permission."
1298                 exit 1
1299         fi

1301         # Set default value for STAFFER, if needed.
1302         if [ -z "$STAFFER" -o "$STAFFER" = "nobody" ]; then
1303                 STAFFER=`/usr/xpg4/bin/id -un`
1304                 export STAFFER
```

```
1305         fi
1306 fi

1308 if [ -z "$MAILTO" -o "$MAILTO" = "nobody" ]; then
1309         MAILTO=$STAFFER
1310         export MAILTO
1311 fi

1313 PATH="$OPTHOME/onbld/bin:$OPTHOME/onbld/bin/${MACH}:/usr/ccs/bin"
1314 PATH="$PATH:$OPTHOME/SUNWspro/bin:$TEAMWARE/bin:/usr/bin:/usr/sbin:/usr/ucb"
1315 PATH="$PATH:/usr/openwin/bin:/usr/sfw/bin:/opt/sfw/bin:."
1316 export PATH

1318 # roots of source trees, both relative to $SRC and absolute.
1319 relsrcdirs="."
1320 abssrcdirs="$SRC"
1404 if [[ -d $CODEMGR_WS/usr/closed && "$CLOSED_IS_PRESENT" != no ]]; then
1405         relsrcdirs="$relsrcdirs ../closed"
1406 fi
1407 abssrcdirs=""
1408 for d in $relsrcdirs; do
1409         abssrcdirs="$abssrcdirs $SRC/$d"
1410 done

1322 unset CH
1323 if [ "$o_FLAG" = "y" ]; then
1324 # root invoked old-style build -- make sure it works as it always has
1325 # by exporting 'CH'.  The current Makefile.master doesn't use this, but
1326 # the old ones still do.
1327         PROTOCMPTERSE="protocmp.terse"
1328         CH=
1329         export CH
1330 else
1331         PROTOCMPTERSE="protocmp.terse -gu"
1332 fi
1333 POUND_SIGN="#"
1334 # have we set RELEASE_DATE in our env file?
1335 if [ -z "$RELEASE_DATE" ]; then
1336         RELEASE_DATE=$(LC_ALL=C date +"%B %Y")
1337 fi
1338 BUILD_DATE=$(LC_ALL=C date +%Y-%b-%d)
1339 BASEWSDIR=$(basename $CODEMGR_WS)
1340 DEV_CM="\"@(#)SunOS Internal Development: $LOGNAME $BUILD_DATE [$BASEWSDIR]\""

1342 # we export POUND_SIGN, RELEASE_DATE and DEV_CM to speed up the build process
1343 # by avoiding repeated shell invocations to evaluate Makefile.master definitions
1344 # we export o_FLAG and X_FLAG for use by makebfu, and by usr/src/pkg/Makefile
1345 export o_FLAG X_FLAG POUND_SIGN RELEASE_DATE DEV_CM

1347 maketype="distributed"
1348 MAKE=dmake
1349 # get the dmake version string alone
1350 DMAKE_VERSION=$( $MAKE -v )
1351 DMAKE_VERSION=${DMAKE_VERSION#*: }
1352 # focus in on just the dotted version number alone
1353 DMAKE_MAJOR=$( echo $DMAKE_VERSION | \
1354         sed -e 's/.*\<\([^.]*\.[^   ]*\).*$/\1/' )
1355 # extract the second (or final) integer
1356 DMAKE_MINOR=${DMAKE_MAJOR#*.}
1357 DMAKE_MINOR=${DMAKE_MINOR%%.*}
1358 # extract the first integer
1359 DMAKE_MAJOR=${DMAKE_MAJOR%%.*}
1360 CHECK_DMAKE=${CHECK_DMAKE:-y}
1361 # x86 was built on the 12th, sparc on the 13th.
1362 if [ "$CHECK_DMAKE" = "y" -a \
1363      "$DMAKE_VERSION" != "Sun Distributed Make 7.3 2003/03/12" -a \
```



```
1364      "$DMAKE_VERSION" != "Sun Distributed Make 7.3 2003/03/13" -a \( \
1365      "$DMAKE_MAJOR" -lt 7 -o \
1366      "$DMAKE_MAJOR" -eq 7 -a "$DMAKE_MINOR" -lt 4 \) ]; then
1367         if [ -z "$DMAKE_VERSION" ]; then
1368                 echo "$MAKE is missing."
1369                 exit 1
1370         fi
1371         echo 'whence $MAKE'" version is:"
1372         echo "  ${DMAKE_VERSION}"
1373         cat <<EOF

1375 This version may not be safe for use.  Either set TEAMWARE to a better
1376 path or (if you really want to use this version of dmake anyway), add
1377 the following to your environment to disable this check:

1379   CHECK_DMAKE=n
1380 EOF
1381         exit 1
1382 fi
1383 export PATH
1384 export MAKE

1476 if [[ "$O_FLAG" = y ]]; then
1477         export TONICBUILD=""
1478 else
1479         export TONICBUILD="#"
1480 fi

1386 if [ "${SUNWSPRO}" != "" ]; then
1387         PATH="${SUNWSPRO}/bin:$PATH"
1388         export PATH
1389 fi

1391 hostname=$(uname -n)
1392 if [[ $DMAKE_MAX_JOBS != +([0-9]) || $DMAKE_MAX_JOBS -eq 0 ]]
1393 then
1394         maxjobs=
1395         if [[ -f $HOME/.make.machines ]]
1396         then
1397                 # Note: there is a hard tab and space character in the []s
1398                 # below.
1399                 egrep -i "^[    ]*$hostname[    \.]" \
1400                         $HOME/.make.machines | read host jobs
1401                 maxjobs=${jobs##*=}
1402         fi

1404         if [[ $maxjobs != +([0-9]) || $maxjobs -eq 0 ]]
1405         then
1406                 # default
1407                 maxjobs=4
1408         fi

1410         export DMAKE_MAX_JOBS=$maxjobs
1411 fi

1413 DMAKE_MODE=parallel;
1414 export DMAKE_MODE

1416 if [ -z "${ROOT}" ]; then
1417         echo "ROOT must be set."
1418         exit 1
1419 fi

1421 #
1422 # if -V flag was given, reset VERSION to V_ARG
1423 #
```

```
1424 if [ "$V_FLAG" = "y" ]; then
1425         VERSION=$V_ARG
1426 fi

1428 #
1429 # Check for IHV root for copying ihv proto area
1430 #
1431 if [ "$X_FLAG" = "y" ]; then
1432         if [ "$IA32_IHV_ROOT" = "" ]; then
1433                 echo "IA32_IHV_ROOT: must be set for copying ihv proto"
1434                 args_ok=n
1435         fi
1436         if [ ! -d "$IA32_IHV_ROOT" ]; then
1437                 echo "$IA32_IHV_ROOT: not found"
1438                 args_ok=n
1439         fi
1440         if [ "$IA32_IHV_WS" = "" ]; then
1441                 echo "IA32_IHV_WS: must be set for copying ihv proto"
1442                 args_ok=n
1443         fi
1444         if [ ! -d "$IA32_IHV_WS" ]; then
1445                 echo "$IA32_IHV_WS: not found"
1446                 args_ok=n
1447         fi
1448 fi

1450 # Append source version
1451 if [ "$SE_FLAG" = "y" ]; then
1452         VERSION="${VERSION}:EXPORT"
1453 fi

1455 if [ "$SD_FLAG" = "y" ]; then
1456         VERSION="${VERSION}:DOMESTIC"
1457 fi

1459 if [ "$SH_FLAG" = "y" ]; then
1460         VERSION="${VERSION}:MODIFIED_SOURCE_PRODUCT"
1461 fi

1463 if [ "$SO_FLAG" = "y" ]; then
1464         VERSION="${VERSION}:OPEN_ONLY"
1465 fi

1467 TMPDIR="/tmp/nightly.tmpdir.$$"
1468 export TMPDIR
1469 rm -rf ${TMPDIR}
1470 mkdir -p $TMPDIR || exit 1
1471 chmod 777 $TMPDIR

1473 #
1474 # Keep elfsign's use of pkcs11_softtoken from looking in the user home
1475 # directory, which doesn't always work.   Needed until all build machines
1476 # have the fix for 6271754
1477 #
1478 SOFTTOKEN_DIR=$TMPDIR
1479 export SOFTTOKEN_DIR

1481 #
1482 # Tools should only be built non-DEBUG.  Keep track of the tools proto
1483 # area path relative to $TOOLS, because the latter changes in an
1484 # export build.
1485 #
1486 # TOOLS_PROTO is included below for builds other than usr/src/tools
1487 # that look for this location.  For usr/src/tools, this will be
1488 # overridden on the $MAKE command line in build_tools().
1489 #
```



```
1490 TOOLS=${SRC}/tools
1491 TOOLS_PROTO_REL=proto/root_${MACH}-nd
1492 TOOLS_PROTO=${TOOLS}/${TOOLS_PROTO_REL};        export TOOLS_PROTO

1494 unset   CFLAGS LD_LIBRARY_PATH LDFLAGS

1496 # create directories that are automatically removed if the nightly script
1497 # fails to start correctly
1498 function newdir {
1499         dir=$1
1500         toadd=
1501         while [ ! -d $dir ]; do
1502                 toadd="$dir $toadd"
1503                 dir=`dirname $dir`
1504         done
1505         torm=
1506         newlist=
1507         for dir in $toadd; do
1508                 if staffer mkdir $dir; then
1509                         newlist="$ISUSER $dir $newlist"
1510                         torm="$dir $torm"
1511                 else
1512                         [ -z "$torm" ] || staffer rmdir $torm
1513                         return 1
1514                 fi
1515         done
1516         newdirlist="$newlist $newdirlist"
1517         return 0
1518 }
______unchanged_portion_omitted_

2051 type bringover_mercurial > /dev/null 2>&1 || function bringover_mercurial {
2052         typeset -x PATH=$PATH

2054         # If the repository doesn't exist yet, then we want to populate it.
2055         if [[ ! -d $CODEMGR_WS/.hg ]]; then
2056                 staffer hg init $CODEMGR_WS
2057                 staffer echo "[paths]" > $CODEMGR_WS/.hg/hgrc
2058                 staffer echo "default=$BRINGOVER_WS" >> $CODEMGR_WS/.hg/hgrc
2059                 touch $TMPDIR/new_repository
2060         fi

2158         #
2159         # If the user set CLOSED_BRINGOVER_WS and didn't set CLOSED_IS_PRESENT
2160         # to "no," then we'll want to initialise the closed repository
2161         #
2162         # We use $orig_closed_is_present instead of $CLOSED_IS_PRESENT,
2163         # because for newly-created source trees, the latter will be "no"
2164         # until after the bringover completes.
2165         #
2166         if [[ "$orig_closed_is_present" != "no" && \
2167             -n "$CLOSED_BRINGOVER_WS" && \
2168             ! -d $CODEMGR_WS/usr/closed/.hg ]]; then
2169                 staffer mkdir -p $CODEMGR_WS/usr/closed
2170                 staffer hg init $CODEMGR_WS/usr/closed
2171                 staffer echo "[paths]" > $CODEMGR_WS/usr/closed/.hg/hgrc
2172                 staffer echo "default=$CLOSED_BRINGOVER_WS" >> $CODEMGR_WS/usr/c
2173                 touch $TMPDIR/new_closed
2174                 export CLOSED_IS_PRESENT=yes
2175         fi

2062         typeset -x HGMERGE="/bin/false"

2064         #
2065         # If the user has changes, regardless of whether those changes are
2066         # committed, and regardless of whether those changes conflict, then
```

```
2067         # we'll attempt to merge them either implicitly (uncommitted) or
2068         # explicitly (committed).
2069         #
2070         # These are the messages we'll use to help clarify mercurial output
2071         # in those cases.
2072         #
2073         typeset mergefailmsg="\
2074 ***\n\
2075 *** nightly was unable to automatically merge your changes.  You should\n\
2076 *** redo the full merge manually, following the steps outlined by mercurial\n\
2077 *** above, then restart nightly.\n\
2078 ***\n"
2079         typeset mergepassmsg="\
2080 ***\n\
2081 *** nightly successfully merged your changes.  This means that your working\n\
2082 *** directory has been updated, but those changes are not yet committed.\n\
2083 *** After nightly completes, you should validate the results of the merge,\n\
2084 *** then use hg commit manually.\n\
2085 ***\n"

2087         #
2088         # For each repository in turn:
2089         #
2090         # 1. Do the pull.  If this fails, dump the output and bail out.
2091         #
2092         # 2. If the pull resulted in an extra head, do an explicit merge.
2093         #    If this fails, dump the output and bail out.
2094         #
2095         # Because we can't rely on Mercurial to exit with a failure code
2096         # when a merge fails (Mercurial issue #186), we must grep the
2097         # output of pull/merge to check for attempted and/or failed merges.
2098         #
2099         # 3. If a merge failed, set the message and fail the bringover.
2100         #
2101         # 4. Otherwise, if a merge succeeded, set the message
2102         #
2103         # 5. Dump the output, and any message from step 3 or 4.
2104         #

2106         typeset HG_SOURCE=$BRINGOVER_WS
2107         if [ ! -f $TMPDIR/new_repository ]; then
2108                 HG_SOURCE=$TMPDIR/open_bundle.hg
2109                 staffer hg --cwd $CODEMGR_WS incoming --bundle $HG_SOURCE \
2110                     -v $BRINGOVER_WS > $TMPDIR/incoming_open.out

2112                 #
2113                 # If there are no incoming changesets, then incoming will
2114                 # fail, and there will be no bundle file.  Reset the source,
2115                 # to allow the remaining logic to complete with no false
2116                 # negatives.  (Unlike incoming, pull will return success
2117                 # for the no-change case.)
2118                 #
2119                 if (( $? != 0 )); then
2120                         HG_SOURCE=$BRINGOVER_WS
2121                 fi
2122         fi

2124         staffer hg --cwd $CODEMGR_WS pull -u $HG_SOURCE \
2125             > $TMPDIR/pull_open.out 2>&1
2126         if (( $? != 0 )); then
2127                 printf "%s: pull failed as follows:\n\n" "$CODEMGR_WS"
2128                 cat $TMPDIR/pull_open.out
2129                 if grep "^merging.*failed" $TMPDIR/pull_open.out > /dev/null 2>&
2130                         printf "$mergefailmsg"
2131                 fi
2132                 touch $TMPDIR/bringover_failed
```



```
2133                 return
2134         fi

2136         if grep "not updating" $TMPDIR/pull_open.out > /dev/null 2>&1; then
2137                 staffer hg --cwd $CODEMGR_WS merge \
2138                     >> $TMPDIR/pull_open.out 2>&1
2139                 if (( $? != 0 )); then
2140                         printf "%s: merge failed as follows:\n\n" \
2141                             "$CODEMGR_WS"
2142                         cat $TMPDIR/pull_open.out
2143                         if grep "^merging.*failed" $TMPDIR/pull_open.out \
2144                             > /dev/null 2>&1; then
2145                                 printf "$mergefailmsg"
2146                         fi
2147                         touch $TMPDIR/bringover_failed
2148                         return
2149                 fi
2150         fi

2152         printf "updated %s with the following results:\n" "$CODEMGR_WS"
2153         cat $TMPDIR/pull_open.out
2154         if grep "^merging" $TMPDIR/pull_open.out >/dev/null 2>&1; then
2155                 printf "$mergepassmsg"
2156         fi
2157         printf "\n"

2159         #
2275         # We only want to update usr/closed if it exists, and we haven't been
2276         # told not to via $CLOSED_IS_PRESENT, and we actually know where to
2277         # pull from ($CLOSED_BRINGOVER_WS).
2278         #
2279         if [[ $CLOSED_IS_PRESENT = yes && \
2280             -d $CODEMGR_WS/usr/closed/.hg && \
2281             -n $CLOSED_BRINGOVER_WS ]]; then

2283                 HG_SOURCE=$CLOSED_BRINGOVER_WS
2284                 if [ ! -f $TMPDIR/new_closed ]; then
2285                         HG_SOURCE=$TMPDIR/closed_bundle.hg
2286                         staffer hg --cwd $CODEMGR_WS/usr/closed incoming \
2287                             --bundle $HG_SOURCE -v $CLOSED_BRINGOVER_WS \
2288                             > $TMPDIR/incoming_closed.out

2290                         #
2291                         # If there are no incoming changesets, then incoming wil
2292                         # fail, and there will be no bundle file.  Reset the sou
2293                         # to allow the remaining logic to complete with no false
2294                         # negatives.  (Unlike incoming, pull will return success
2295                         # for the no-change case.)
2296                         #
2297                         if (( $? != 0 )); then
2298                                 HG_SOURCE=$CLOSED_BRINGOVER_WS
2299                         fi
2300                 fi

2302                 staffer hg --cwd $CODEMGR_WS/usr/closed pull -u \
2303                         $HG_SOURCE > $TMPDIR/pull_closed.out 2>&1
2304                 if (( $? != 0 )); then
2305                         printf "closed pull failed as follows:\n\n"
2306                         cat $TMPDIR/pull_closed.out
2307                         if grep "^merging.*failed" $TMPDIR/pull_closed.out \
2308                             > /dev/null 2>&1; then
2309                                 printf "$mergefailmsg"
2310                         fi
2311                         touch $TMPDIR/bringover_failed
2312                         return
2313                 fi
```

```
2315                 if grep "not updating" $TMPDIR/pull_closed.out > /dev/null 2>&1;
2316                         staffer hg --cwd $CODEMGR_WS/usr/closed merge \
2317                             >> $TMPDIR/pull_closed.out 2>&1
2318                         if (( $? != 0 )); then
2319                                 printf "closed merge failed as follows:\n\n"
2320                                 cat $TMPDIR/pull_closed.out
2321                                 if grep "^merging.*failed" $TMPDIR/pull_closed.o
2322                                         printf "$mergefailmsg"
2323                                 fi
2324                                 touch $TMPDIR/bringover_failed
2325                                 return
2326                         fi
2327                 fi

2329                 printf "updated %s with the following results:\n" \
2330                     "$CODEMGR_WS/usr/closed"
2331                 cat $TMPDIR/pull_closed.out
2332                 if grep "^merging" $TMPDIR/pull_closed.out > /dev/null 2>&1; the
2333                         printf "$mergepassmsg"
2334                 fi
2335         fi

2337         #
2160         # Per-changeset output is neither useful nor manageable for a
2161         # newly-created repository.
2162         #
2163         if [ -f $TMPDIR/new_repository ]; then
2164                 return
2165         fi

2167         printf "\nadded the following changesets to open repository:\n"
2168         cat $TMPDIR/incoming_open.out

2170         #
2171         # The closed repository could have been newly created, even though
2172         # the open one previously existed...
2173         #
2174         if [ -f $TMPDIR/new_closed ]; then
2175                 return
2176         fi

2178         if [ -f $TMPDIR/incoming_closed.out ]; then
2179                 printf "\nadded the following changesets to closed repository:\n
2180                 cat $TMPDIR/incoming_closed.out
2181         fi
2182 }
```

_____***unchanged_portion_omitted_***

```
2219 #
2220 #       Decide whether to bringover to the codemgr workspace
2221 #
2222 if [ "$n_FLAG" = "n" ]; then
2223         PARENT_SCM_TYPE=$(parent_wstype)

2225         if [[ $SCM_TYPE != none && $SCM_TYPE != $PARENT_SCM_TYPE ]]; then
2226                 echo "cannot bringover from $PARENT_SCM_TYPE to $SCM_TYPE, " \
2227                     "quitting at 'date'." | tee -a $mail_msg_file >> $LOGFILE
2228                 exit 1
2229         fi

2231         run_hook PRE_BRINGOVER

2233         echo "\n==== bringover to $CODEMGR_WS at 'date' ====\n" >> $LOGFILE
2234         echo "\n==== BRINGOVER LOG ====\n" >> $mail_msg_file
```



```
2236         eval "bringover_${PARENT_SCM_TYPE}" 2>&1 |
2237                 tee -a $mail_msg_file >> $LOGFILE

2239         if [ -f $TMPDIR/bringover_failed ]; then
2240                 rm -f $TMPDIR/bringover_failed
2241                 build_ok=n
2242                 echo "trouble with bringover, quitting at 'date'." |
2243                         tee -a $mail_msg_file >> $LOGFILE
2244                 exit 1
2245         fi

2247         #
2248         # It's possible that we used the bringover above to create
2249         # $CODEMGR_WS.  If so, then SCM_TYPE was previously "none,"
2250         # but should now be the same as $BRINGOVER_WS.
2251         #
2252         [[ $SCM_TYPE = none ]] && SCM_TYPE=$PARENT_SCM_TYPE

2254         run_hook POST_BRINGOVER

2434         #
2435         # Possible transition from pre-split workspace to split
2436         # workspace.  See if the bringover changed anything.
2437         #
2438         CLOSED_IS_PRESENT="$orig_closed_is_present"
2256         check_closed_tree

2258 else
2259         echo "\n==== No bringover to $CODEMGR_WS ====\n" >> $LOGFILE
2260 fi

2262 if [[ "$O_FLAG" = y ]]; then
2445 if [[ "$O_FLAG" = y && "$CLOSED_IS_PRESENT" != "yes" ]]; then
2263         build_ok=n
2264         echo "OpenSolaris binary deliverables need usr/closed." \
2265             | tee -a "$mail_msg_file" >> $LOGFILE
2266         exit 1
2267 fi

2269 # Safeguards
2270 [[ -v CODEMGR_WS ]] || fatal_error "Error: Variable CODEMGR_WS not set."
2271 [[ -d "${CODEMGR_WS}" ]] || fatal_error "Error: ${CODEMGR_WS} is not a directory
2272 [[ -f "${CODEMGR_WS}/usr/src/Makefile" ]] || fatal_error "Error: ${CODEMGR_WS}/u

2274 echo "\n==== Build environment ====\n" | tee -a $build_environ_file >> $LOGFILE

2276 # System
2277 whence uname | tee -a $build_environ_file >> $LOGFILE
2278 uname -a 2>&1 | tee -a $build_environ_file >> $LOGFILE
2279 echo | tee -a $build_environ_file >> $LOGFILE

2281 # make
2282 whence $MAKE | tee -a $build_environ_file >> $LOGFILE
2283 $MAKE -v | tee -a $build_environ_file >> $LOGFILE
2284 echo "number of concurrent jobs = $DMAKE_MAX_JOBS" |
2285     tee -a $build_environ_file >> $LOGFILE

2287 #
2288 # Report the compiler versions.
2289 #

2291 if [[ ! -f $SRC/Makefile ]]; then
2292         build_ok=n
2293         echo "\nUnable to find \"Makefile\" in $SRC." | \
2294             tee -a $build_environ_file >> $LOGFILE
2295         exit 1
```

```
2296 fi

2298 ( cd $SRC
2299   for target in cc-version cc64-version java-version; do
2300         echo
2301         #
2302         # Put statefile somewhere we know we can write to rather than trip
2303         # over a read-only $srcroot.
2304         #
2305         rm -f $TMPDIR/make-state
2306         export SRC
2307         if $MAKE -K $TMPDIR/make-state -e $target 2>/dev/null; then
2308                 continue
2309         fi
2310         touch $TMPDIR/nocompiler
2311   done
2312   echo
2313 ) | tee -a $build_environ_file >> $LOGFILE

2315 if [ -f $TMPDIR/nocompiler ]; then
2316         rm -f $TMPDIR/nocompiler
2317         build_ok=n
2318         echo "Aborting due to missing compiler." |
2319                 tee -a $build_environ_file >> $LOGFILE
2320         exit 1
2321 fi

2323 # as
2324 whence as | tee -a $build_environ_file >> $LOGFILE
2325 as -V 2>&1 | head -1 | tee -a $build_environ_file >> $LOGFILE
2326 echo | tee -a $build_environ_file >> $LOGFILE

2328 # Check that we're running a capable link-editor
2329 whence ld | tee -a $build_environ_file >> $LOGFILE
2330 LDVER=`ld -V 2>&1`
2331 echo $LDVER | tee -a $build_environ_file >> $LOGFILE
2332 LDVER=`echo $LDVER | sed -e "s/.*-1\.\([0-9]*\).*/\1/"`
2333 if [ `expr $LDVER \< 422` -eq 1 ]; then
2334         echo "The link-editor needs to be at version 422 or higher to build" | \
2335             tee -a $build_environ_file >> $LOGFILE
2336         echo "the latest stuff.  Hope your build works." | \
2337             tee -a $build_environ_file >> $LOGFILE
2338 fi

2340 #
2341 # Build and use the workspace's tools if requested
2342 #
2343 if [[ "$t_FLAG" = "y" || "$O_FLAG" = y ]]; then
2344         set_non_debug_build_flags

2346         build_tools ${TOOLS_PROTO}
2347         if [[ $? != 0  && "$t_FLAG" = y ]]; then
2348                 use_tools $TOOLS_PROTO
2349         fi
2350 fi

2352 #
2353 # copy ihv proto area in addition to the build itself
2354 #
2355 if [ "$X_FLAG" = "y" ]; then
2356         copy_ihv_proto
2357 fi

2359 if [ "$i_FLAG" = "y" -a "$SH_FLAG" = "y" ]; then
2360         echo "\n==== NOT Building base OS-Net source ====\n" | \
2361             tee -a $LOGFILE >> $mail_msg_file
```



```
2362 else
2363         # timestamp the start of the normal build; the findunref tool uses it.
2364         touch $SRC/.build.tstamp

2366         normal_build
2367 fi

2369 #
2370 # Generate the THIRDPARTYLICENSE files if needed.  This is done after
2371 # the build, so that dynamically-created license files are there.
2372 # It's done before findunref to help identify license files that need
2373 # to be added to tools/opensolaris/license-list.
2374 #
2375 if [ "$O_FLAG" = y -a "$build_ok" = y ]; then
2376         echo "\n==== Generating THIRDPARTYLICENSE files ====\n" |
2377             tee -a "$mail_msg_file" >> "$LOGFILE"

2379         if [ -d $ROOT/licenses/usr ]; then
2380                 ( cd $ROOT/licenses ; \
2381                     mktpl $SRC/pkg/license-list ) >> "$LOGFILE" 2>&1
2382                 if (( $? != 0 )) ; then
2383                         echo "Couldn't create THIRDPARTYLICENSE files" |
2384                             tee -a "$mail_msg_file" >> "$LOGFILE"
2385                 fi
2386         else
2387                 echo "No licenses found under $ROOT/licenses" |
2388                     tee -a "$mail_msg_file" >> "$LOGFILE"
2389         fi
2390 fi

2392 ORIG_SRC=$SRC
2393 BINARCHIVE=${CODEMGR_WS}/bin-${MACH}.cpio.Z

2395 if [ "$SE_FLAG" = "y" -o "$SD_FLAG" = "y" -o "$SH_FLAG" = "y" ]; then
2396         save_binaries
2397 fi


2400 # EXPORT_SRC comes after CRYPT_SRC since a domestic build will need
2401 # $SRC pointing to the export_source usr/src.

2403 if [ "$SE_FLAG" = "y" -o "$SD_FLAG" = "y" -o "$SH_FLAG" = "y" ]; then
2404         if [ "$SD_FLAG" = "y" -a $build_ok = y ]; then
2405             set_up_source_build ${CODEMGR_WS} ${CRYPT_SRC} CRYPT_SRC
2406         fi

2408         if [ $build_ok = y ]; then
2409             set_up_source_build ${CODEMGR_WS} ${EXPORT_SRC} EXPORT_SRC
2410         fi
2411 fi

2413 if [ "$SD_FLAG" = "y" -a $build_ok = y ]; then
2414         # drop the crypt files in place.
2415         cd ${EXPORT_SRC}
2416         echo "\nextracting crypt_files.cpio.Z onto export_source.\n" \
2417             >> ${LOGFILE}
2418         zcat ${CODEMGR_WS}/crypt_files.cpio.Z | \
2419             cpio -idmucvB 2>/dev/null >> ${LOGFILE}
2420         if [ "$?" = "0" ]; then
2421                 echo "\n==== DOMESTIC extraction succeeded ====\n" \
2422                     >> $mail_msg_file
2423         else
2424                 echo "\n==== DOMESTIC extraction failed ====\n" \
2425                     >> $mail_msg_file
2426         fi
```

```
2428 fi

2430 if [ "$SO_FLAG" = "y" -a "$build_ok" = y ]; then
2431         #
2432         # Copy the open sources into their own tree.
2433         # If copy_source fails, it will have already generated an
2434         # error message and set build_ok=n, so we don't need to worry
2435         # about that here.
2436         #
2437         copy_source $CODEMGR_WS $OPEN_SRCDIR OPEN_SOURCE usr/src
2438 fi

2440 if [ "$SO_FLAG" = "y" -a "$build_ok" = y ]; then
2441         SRC=$OPEN_SRCDIR/usr/src
2625         export CLOSED_IS_PRESENT=no
2442 fi

2444 if is_source_build && [ $build_ok = y ] ; then
2445         # remove proto area(s) here, since we don't clobber
2446         rm -rf 'allprotos'
2447         if [ "$t_FLAG" = "y" ]; then
2448                 set_non_debug_build_flags
2449                 ORIG_TOOLS=$TOOLS
2450                 #
2451                 # SRC was set earlier to point to the source build
2452                 # source tree (e.g., $EXPORT_SRC).
2453                 #
2454                 TOOLS=${SRC}/tools
2455                 TOOLS_PROTO=${TOOLS}/${TOOLS_PROTO_REL}; export TOOLS_PROTO
2456                 build_tools ${TOOLS_PROTO}
2457                 if [[ $? != 0 ]]; then
2458                         use_tools ${TOOLS_PROTO}
2459                 fi
2460         fi

2462         normal_build
2463 fi

2465 #
2466 # There are several checks that need to look at the proto area, but
2467 # they only need to look at one, and they don't care whether it's
2468 # DEBUG or non-DEBUG.
2469 #
2470 if [[ "$MULTI_PROTO" = yes && "$D_FLAG" = n ]]; then
2471         checkroot=$ROOT-nd
2472 else
2473         checkroot=$ROOT
2474 fi

2476 if [ "$build_ok" = "y" ]; then
2477         echo "\n==== Creating protolist system file at 'date' ====" \
2478                 >> $LOGFILE
2479         protolist $checkroot > $ATLOG/proto_list_${MACH}
2480         echo "==== protolist system file created at 'date' ====\n" \
2481                 >> $LOGFILE

2483         if [ "$N_FLAG" != "y" ]; then

2485                 E1=
2486                 f1=
2487                 if [ -d "$SRC/pkgdefs" ]; then
2488                         f1="$SRC/pkgdefs/etc/exception_list_$MACH"
2489                         if [ "$X_FLAG" = "y" ]; then
2490                                 f1="$f1 $IA32_IHV_WS/usr/src/pkgdefs/etc/excepti
2491                         fi
2492                 fi
```



```
2494                 for f in $f1; do
2495                         if [ -f "$f" ]; then
2496                                 E1="$E1 -e $f"
2497                         fi
2498                 done

2500                 E2=
2501                 f2=
2502                 if [ -d "$SRC/pkg" ]; then
2503                         f2="$f2 exceptions/packaging"
2504                 fi

2506                 for f in $f2; do
2507                         if [ -f "$f" ]; then
2508                                 E2="$E2 -e $f"
2509                         fi
2510                 done

2512                 if [ -f "$REF_PROTO_LIST" ]; then
2513                         #
2514                         # For builds that copy the IHV proto area (-X), add the
2515                         # IHV proto list to the reference list if the reference
2516                         # was built without -X.
2517                         #
2518                         # For builds that don't copy the IHV proto area, add the
2519                         # IHV proto list to the build's proto list if the
2520                         # reference was built with -X.
2521                         #
2522                         # Use the presence of the first file entry of the cached
2523                         # IHV proto list in the reference list to determine
2524                         # whether it was built with -X or not.
2525                         #
2526                         IHV_REF_PROTO_LIST=$SRC/pkg/proto_list_ihv_$MACH
2527                         grepfor=$(nawk '$1 == "f" { print $2; exit }' \
2528                                 $IHV_REF_PROTO_LIST 2> /dev/null)
2529                         if [ $? = 0 -a -n "$grepfor" ]; then
2530                                 if [ "$X_FLAG" = "y" ]; then
2531                                         grep -w "$grepfor" \
2532                                                 $REF_PROTO_LIST > /dev/null
2533                                         if [ ! "$?" = "0" ]; then
2534                                                 REF_IHV_PROTO="-d $IHV_REF_PROTO
2535                                         fi
2536                                 else
2537                                         grep -w "$grepfor" \
2538                                                 $REF_PROTO_LIST > /dev/null
2539                                         if [ "$?" = "0" ]; then
2540                                                 IHV_PROTO_LIST="$IHV_REF_PROTO_L
2541                                         fi
2542                                 fi
2543                         fi
2544                 fi
2545         fi

2547         if [ "$N_FLAG" != "y" -a -f $SRC/pkgdefs/Makefile ]; then
2548                 echo "\n==== Impact on SVr4 packages ====\n" >> $mail_msg_file
2549                 #
2550                 # Compare the build's proto list with current package
2551                 # definitions to audit the quality of package
2552                 # definitions and makefile install targets. Use the
2553                 # current exception list.
2554                 #
2555                 PKGDEFS_LIST=""
2556                 for d in $abssrcdirs; do
2557                         if [ -d $d/pkgdefs ]; then
2558                                 PKGDEFS_LIST="$PKGDEFS_LIST -d $d/pkgdefs"
```

```
2559                            fi
2560                    done
2561                    if [ "$X_FLAG" = "y" -a \
2562                        -d $IA32_IHV_WS/usr/src/pkgdefs ]; then
2563                            PKGDEFS_LIST="$PKGDEFS_LIST -d $IA32_IHV_WS/usr/src/pkgd
2564                    fi
2565                    $PROTOCMPTERSE \
2566                        "Files missing from the proto area:" \
2567                        "Files missing from packages:" \
2568                        "Inconsistencies between pkgdefs and proto area:" \
2569                        ${E1} \
2570                        ${PKGDEFS_LIST} \
2571                        $ATLOG/proto_list_${MACH} \
2572                        >> $mail_msg_file
2573            fi

2575            if [ "$N_FLAG" != "y" -a -d $SRC/pkg ]; then
2576                    echo "\n==== Validating manifests against proto area ====\n" \
2577                        >> $mail_msg_file
2578                    ( cd $SRC/pkg ; $MAKE -e protocmp ROOT="$checkroot" ) \
2579                        >> $mail_msg_file

2581            fi

2583            if [ "$N_FLAG" != "y" -a -f "$REF_PROTO_LIST" ]; then
2584                    echo "\n==== Impact on proto area ====\n" >> $mail_msg_file
2585                    if [ -n "$E2" ]; then
2586                            ELIST=$E2
2587                    else
2588                            ELIST=$E1
2589                    fi
2590                    $PROTOCMPTERSE \
2591                            "Files in yesterday's proto area, but not today's:" \
2592                            "Files in today's proto area, but not yesterday's:" \
2593                            "Files that changed between yesterday and today:" \
2594                            ${ELIST} \
2595                            -d $REF_PROTO_LIST \
2596                            $REF_IHV_PROTO \
2597                            $ATLOG/proto_list_${MACH} \
2598                            $IHV_PROTO_LIST \
2599                            >> $mail_msg_file
2600            fi
2601    fi

2603    if [ "$u_FLAG" = "y"  -a "$build_ok" = "y" ]; then
2604            staffer cp $ATLOG/proto_list_${MACH} \
2605                    $PARENT_WS/usr/src/proto_list_${MACH}
2606    fi

2608    # Update parent proto area if necessary. This is done now
2609    # so that the proto area has either DEBUG or non-DEBUG kernels.
2610    # Note that this clears out the lock file, so we can dispense with
2611    # the variable now.
2612    if [ "$U_FLAG" = "y" -a "$build_ok" = "y" ]; then
2613            echo "\n==== Copying proto area to $NIGHTLY_PARENT_ROOT ====\n" | \
2614                tee -a $LOGFILE >> $mail_msg_file
2615            rm -rf $NIGHTLY_PARENT_ROOT/*
2616            unset Ulockfile
2617            mkdir -p $NIGHTLY_PARENT_ROOT
2618            if [[ "$MULTI_PROTO" = no || "$D_FLAG" = y ]]; then
2619                    ( cd $ROOT; tar cf - . |
2620                        ( cd $NIGHTLY_PARENT_ROOT;  umask 0; tar xpf - ) ) 2>&1 |
2621                        tee -a $mail_msg_file >> $LOGFILE
2622            fi
2623            if [[ "$MULTI_PROTO" = yes && "$F_FLAG" = n ]]; then
2624                    rm -rf $NIGHTLY_PARENT_ROOT-nd/*
```



```
2625                    mkdir -p $NIGHTLY_PARENT_ROOT-nd
2626                    cd $ROOT-nd
2627                    ( tar cf - . |
2628                        ( cd $NIGHTLY_PARENT_ROOT-nd; umask 0; tar xpf - ) ) 2>&1 |
2629                        tee -a $mail_msg_file >> $LOGFILE
2630            fi
2631            if [ -n "${NIGHTLY_PARENT_TOOLS_ROOT}" ]; then
2632                    echo "\n==== Copying tools proto area to $NIGHTLY_PARENT_TOOLS_R
2633                        tee -a $LOGFILE >> $mail_msg_file
2634                    rm -rf $NIGHTLY_PARENT_TOOLS_ROOT/*
2635                    mkdir -p $NIGHTLY_PARENT_TOOLS_ROOT
2636                    if [[ "$MULTI_PROTO" = no || "$D_FLAG" = y ]]; then
2637                            ( cd $TOOLS_PROTO; tar cf - . |
2638                                ( cd $NIGHTLY_PARENT_TOOLS_ROOT;
2639                                umask 0; tar xpf - ) ) 2>&1 |
2640                                tee -a $mail_msg_file >> $LOGFILE
2641                    fi
2642            fi
2643    fi

2645    #
2646    # ELF verification: ABI (-A) and runtime (-r) checks
2647    #
2648    if [[ ($build_ok = y) && ( ($A_FLAG = y) || ($r_FLAG = y) ) ]]; then
2649            # Directory ELF-data.$MACH holds the files produced by these tests.
2650            elf_ddir=$SRC/ELF-data.$MACH

2652            # If there is a previous ELF-data backup directory, remove it. Then,
2653            # rotate current ELF-data directory into its place and create a new
2654            # empty directory
2655            rm -rf $elf_ddir.ref
2656            if [[ -d $elf_ddir ]]; then
2657                    mv $elf_ddir $elf_ddir.ref
2658            fi
2659            mkdir -p $elf_ddir

2661            # Call find_elf to produce a list of the ELF objects in the proto area.
2662            # This list is passed to check_rtime and interface_check, preventing
2663            # them from separately calling find_elf to do the same work twice.
2664            find_elf -fr $checkroot > $elf_ddir/object_list

2666            if [[ $A_FLAG = y ]]; then
2667                    echo "\n==== Check versioning and ABI information ====\n"  | \
2668                        tee -a $LOGFILE >> $mail_msg_file

2670                    # Produce interface description for the proto. Report errors.
2671                    interface_check -o -w $elf_ddir -f object_list \
2672                            -i interface -E interface.err
2673                    if [[ -s $elf_ddir/interface.err ]]; then
2674                            tee -a $LOGFILE < $elf_ddir/interface.err \
2675                                    >> $mail_msg_file
2676                    fi

2678                    # If ELF_DATA_BASELINE_DIR is defined, compare the new interface
2679                    # description file to that from the baseline gate. Issue a
2680                    # warning if the baseline is not present, and keep going.
2681                    if [[ "$ELF_DATA_BASELINE_DIR" != '' ]]; then
2682                            base_ifile="$ELF_DATA_BASELINE_DIR/interface"

2684                            echo "\n==== Compare versioning and ABI information" \
2685                                "to baseline ====\n"  | \
2686                                tee -a $LOGFILE >> $mail_msg_file
2687                            echo "Baseline:  $base_ifile\n" >> $LOGFILE

2689                            if [[ -f $base_ifile ]]; then
2690                                    interface_cmp -d -o $base_ifile \
```

```
2691                                    $elf_ddir/interface > $elf_ddir/interface.cm
2692                                if [[ -s $elf_ddir/interface.cmp ]]; then
2693                                        echo | tee -a $LOGFILE >> $mail_msg_file
2694                                        tee -a $LOGFILE < \
2695                                            $elf_ddir/interface.cmp \
2696                                            >> $mail_msg_file
2697                                fi
2698                        else
2699                                echo "baseline not available. comparison" \
2700                                    "skipped" | \
2701                                    tee -a $LOGFILE >> $mail_msg_file
2702                        fi

2704                fi
2705        fi

2707        if [[ $r_FLAG = y ]]; then
2708                echo "\n==== Check ELF runtime attributes ====\n" | \
2709                    tee -a $LOGFILE >> $mail_msg_file

2711                # If we're doing a DEBUG build the proto area will be left
2712                # with debuggable objects, thus don't assert -s.
2713                if [[ $D_FLAG = y ]]; then
2714                        rtime_sflag=""
2715                else
2716                        rtime_sflag="-s"
2717                fi
2718                check_rtime -i -m -v $rtime_sflag -o -w $elf_ddir \
2719                        -D object_list  -f object_list -E runtime.err \
2720                        -I runtime.attr.raw

2722                # check_rtime -I output needs to be sorted in order to
2723                # compare it to that from previous builds.
2724                sort $elf_ddir/runtime.attr.raw > $elf_ddir/runtime.attr
2725                rm $elf_ddir/runtime.attr.raw

2727                # Report errors
2728                if [[ -s $elf_ddir/runtime.err ]]; then
2729                        tee -a $LOGFILE < $elf_ddir/runtime.err \
2730                                >> $mail_msg_file
2731                fi

2733                # If there is an ELF-data directory from a previous build,
2734                # then diff the attr files. These files contain information
2735                # about dependencies, versioning, and runpaths. There is some
2736                # overlap with the ABI checking done above, but this also
2737                # flushes out non-ABI interface differences along with the
2738                # other information.
2739                echo "\n==== Diff ELF runtime attributes" \
2740                    "(since last build) ====\n" | \
2741                    tee -a $LOGFILE >> $mail_msg_file >> $mail_msg_file

2743                if [[ -f $elf_ddir.ref/runtime.attr ]]; then
2744                        diff $elf_ddir.ref/runtime.attr \
2745                                $elf_ddir/runtime.attr \
2746                                >> $mail_msg_file
2747                fi
2748        fi

2750        # If -u set, copy contents of ELF-data.$MACH to the parent workspace.
2751        if [[ "$u_FLAG" = "y" ]]; then
2752                p_elf_ddir=$PARENT_WS/usr/src/ELF-data.$MACH

2754                # If parent lacks the ELF-data.$MACH directory, create it
2755                if [[ ! -d $p_elf_ddir ]]; then
2756                        staffer mkdir -p $p_elf_ddir
```



```
2757                fi

2759                # These files are used asynchronously by other builds for ABI
2760                # verification, as above for the -A option. As such, we require
2761                # the file replacement to be atomic. Copy the data to a temp
2762                # file in the same filesystem and then rename into place.
2763                (
2764                        cd $elf_ddir
2765                        for elf_dfile in *; do
2766                                staffer cp $elf_dfile \
2767                                    ${p_elf_ddir}/${elf_dfile}.new
2768                                staffer mv -f ${p_elf_ddir}/${elf_dfile}.new \
2769                                    ${p_elf_ddir}/${elf_dfile}
2770                        done
2771                )
2772        fi
2773 fi

2775 # DEBUG lint of kernel begins

2777 if [ "$i_CMD_LINE_FLAG" = "n" -a "$l_FLAG" = "y" ]; then
2778         if [ "$LINTDIRS" = "" ]; then
2779                 # LINTDIRS="$SRC/uts y $SRC/stand y $SRC/psm y"
2780                 LINTDIRS="$SRC y"
2781         fi
2782         set $LINTDIRS
2783         while [ $# -gt 0 ]; do
2784                 dolint $1 $2; shift; shift
2785         done
2786 else
2787         echo "\n==== No '$MAKE lint' ====\n" >> $LOGFILE
2788 fi

2790 # "make check" begins

2792 if [ "$i_CMD_LINE_FLAG" = "n" -a "$C_FLAG" = "y" ]; then
2793         # remove old check.out
2794         rm -f $SRC/check.out

2796         rm -f $SRC/check-${MACH}.out
2797         cd $SRC
2798         $MAKE -ek check ROOT="$checkroot" 2>&1 | tee -a $SRC/check-${MACH}.out \
2799             >> $LOGFILE
2800         echo "\n==== cstyle/hdrchk errors ====\n" >> $mail_msg_file

2802         grep ":" $SRC/check-${MACH}.out |
2803                 egrep -v "Ignoring unknown host" | \
2804                 sort | uniq >> $mail_msg_file
2805 else
2806         echo "\n==== No '$MAKE check' ====\n" >> $LOGFILE
2807 fi

2809 echo "\n==== Find core files ====\n" | \
2810     tee -a $LOGFILE >> $mail_msg_file

2812 find $abssrcdirs -name core -a -type f -exec file {} \; | \
2813         tee -a $LOGFILE >> $mail_msg_file

2815 if [ "$f_FLAG" = "y" -a "$build_ok" = "y" ]; then
2816         echo "\n==== Diff unreferenced files (since last build) ====\n" \
2817             | tee -a $LOGFILE >>$mail_msg_file
2818         rm -f $SRC/unref-${MACH}.ref
2819         if [ -f $SRC/unref-${MACH}.out ]; then
2820                 mv $SRC/unref-${MACH}.out $SRC/unref-${MACH}.ref
2821         fi
```

```
2823         findunref -S $SCM_TYPE -t $SRC/.build.tstamp -s usr $CODEMGR_WS \
2824             ${TOOLS}/findunref/exception_list 2>> $mail_msg_file | \
2825             sort > $SRC/unref-${MACH}.out

2827         if [ ! -f $SRC/unref-${MACH}.ref ]; then
2828                 cp $SRC/unref-${MACH}.out $SRC/unref-${MACH}.ref
2829         fi

2831         diff $SRC/unref-${MACH}.ref $SRC/unref-${MACH}.out >>$mail_msg_file
2832 fi

2834 #
2835 # Generate the OpenSolaris deliverables if requested.  Some of these
2836 # steps need to come after findunref and are commented below.
2837 #

2839 # If we are doing an OpenSolaris _source_ build (-S O) then we do
2840 # not have usr/closed available to us to generate closedbins from,
2841 # so skip this part.
2842 if [ "$SO_FLAG" = n -a "$O_FLAG" = y -a "$build_ok" = y ]; then
2843         echo "\n==== Generating OpenSolaris tarballs ====\n" | \
2844             tee -a $mail_msg_file >> $LOGFILE

2846         cd $CODEMGR_WS

2848         #
2849         # This step grovels through the package manifests, so it
2850         # must come after findunref.
2851         #
2852         # We assume no DEBUG vs non-DEBUG package content variation
2853         # here; if that changes, then the "make all" in $SRC/pkg will
2854         # need to be moved into the conditionals and repeated for each
2855         # different build.
2856         #
2857         echo "Generating closed binaries tarball(s)..." >> $LOGFILE
2858         closed_basename=on-closed-bins
2859         if [ "$D_FLAG" = y ]; then
2860                 bindrop "$closed_basename" >>"$LOGFILE" 2>&1
2861                 if (( $? != 0 )) ; then
2862                         echo "Couldn't create DEBUG closed binaries." |
2863                             tee -a $mail_msg_file >> $LOGFILE
2864                         build_ok=n
2865                 fi
2866         fi
2867         if [ "$F_FLAG" = n ]; then
2868                 bindrop -n "$closed_basename-nd" >>"$LOGFILE" 2>&1
2869                 if (( $? != 0 )) ; then
2870                         echo "Couldn't create non-DEBUG closed binaries." |
2871                             tee -a $mail_msg_file >> $LOGFILE
2872                         build_ok=n
2873                 fi
2874         fi

2876         echo "Generating README.opensolaris..." >> $LOGFILE
2877         cat $SRC/tools/opensolaris/README.opensolaris.tmpl | \
2878             mkreadme_osol $CODEMGR_WS/README.opensolaris >> $LOGFILE 2>&1
2879         if (( $? != 0 )) ; then
2880                 echo "Couldn't create README.opensolaris." |
2881                     tee -a $mail_msg_file >> $LOGFILE
2882                 build_ok=n
2883         fi
2884 fi

2886 # Verify that the usual lists of files, such as exception lists,
2887 # contain only valid references to files.  If the build has failed,
2888 # then don't check the proto area.
```



```
2889 CHECK_PATHS=${CHECK_PATHS:-y}
2890 if [ "$CHECK_PATHS" = y -a "$N_FLAG" != y ]; then
2891         echo "\n==== Check lists of files ====\n" | tee -a $LOGFILE \
2892                 >>$mail_msg_file
2893         arg=-b
2894         [ "$build_ok" = y ] && arg=
2895         checkpaths $arg $checkroot 2>&1 | tee -a $LOGFILE >>$mail_msg_file
2896 fi

2898 if [ "$M_FLAG" != "y" -a "$build_ok" = y ]; then
2899         echo "\n==== Impact on file permissions ====\n" \
2900                 >> $mail_msg_file

2902         abspkgdefs=
2903         abspkg=
2904         for d in $abssrcdirs; do
2905                 if [ -d "$d/pkgdefs" ]; then
2906                         abspkgdefs="$abspkgdefs $d"
2907                 fi
2908                 if [ -d "$d/pkg" ]; then
2909                         abspkg="$abspkg $d"
2910                 fi
2911         done

2913         if [ -n "$abspkgdefs" ]; then
2914                 pmodes -qvdP \
2915                     'find $abspkgdefs -name pkginfo.tmpl -print -o \
2916                     -name .del\* -prune | sed -e 's:/pkginfo.tmpl$::' | \
2917                     sort -u' >> $mail_msg_file
2918         fi

2920         if [ -n "$abspkg" ]; then
2921                 for d in "$abspkg"; do
2922                         ( cd $d/pkg ; $MAKE -e pmodes ) >> $mail_msg_file
2923                 done
2924         fi
2925 fi

2927 if [ "$w_FLAG" = "y" -a "$build_ok" = "y" ]; then
2928         if [[ "$MULTI_PROTO" = no || "$D_FLAG" = y ]]; then
2929                 do_wsdiff DEBUG $ROOT.prev $ROOT
2930         fi

2932         if [[ "$MULTI_PROTO" = yes && "$F_FLAG" = n ]]; then
2933                 do_wsdiff non-DEBUG $ROOT-nd.prev $ROOT-nd
2934         fi
2935 fi

2937 END_DATE=`date`
2938 echo "==== Nightly $maketype build completed: $END_DATE ====" | \
2939     tee -a $LOGFILE >> $build_time_file

2941 typeset -i10 hours
2942 typeset -Z2 minutes
2943 typeset -Z2 seconds

2945 elapsed_time=$SECONDS
2946 ((hours = elapsed_time / 3600 ))
2947 ((minutes = elapsed_time / 60  % 60))
2948 ((seconds = elapsed_time % 60))

2950 echo "\n==== Total build time ====" | \
2951     tee -a $LOGFILE >> $build_time_file
2952 echo "\nreal    ${hours}:${minutes}:${seconds}" | \
2953     tee -a $LOGFILE >> $build_time_file
```

```
2955 if [ "$u_FLAG" = "y" -a "$f_FLAG" = "y" -a "$build_ok" = "y" ]; then
2956         staffer cp ${SRC}/unref-${MACH}.out $PARENT_WS/usr/src/

2958         #
2959         # Produce a master list of unreferenced files -- ideally, we'd
2960         # generate the master just once after all of the nightlies
2961         # have finished, but there's no simple way to know when that
2962         # will be.  Instead, we assume that we're the last nightly to
2963         # finish and merge all of the unref-${MACH}.out files in
2964         # $PARENT_WS/usr/src/.  If we are in fact the final ${MACH} to
2965         # finish, then this file will be the authoritative master
2966         # list.  Otherwise, another ${MACH}'s nightly will eventually
2967         # overwrite ours with its own master, but in the meantime our
2968         # temporary "master" will be no worse than any older master
2969         # which was already on the parent.
2970         #

2972         set -- $PARENT_WS/usr/src/unref-*.out
2973         cp "$1" ${TMPDIR}/unref.merge
2974         shift

2976         for unreffile; do
2977                 comm -12 ${TMPDIR}/unref.merge "$unreffile" > ${TMPDIR}/unref.$$
2978                 mv ${TMPDIR}/unref.$$ ${TMPDIR}/unref.merge
2979         done

2981         staffer cp ${TMPDIR}/unref.merge $PARENT_WS/usr/src/unrefmaster.out
2982 fi

2984 #
2985 # All done save for the sweeping up.
2986 # (whichever exit we hit here will trigger the "cleanup" trap which
2987 # optionally sends mail on completion).
2988 #
2989 if [ "$build_ok" = "y" ]; then
2990         exit 0
2991 fi
2992 exit 1
```

```
**********************************************************
   10351 Wed Oct 16 17:41:08 2013
new/usr/src/tools/scripts/ws.sh
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
______unchanged_portion_omitted_

 109 if [[ "$1" = "-e" ]]; then
 110         setenv=true
 111         shift
 112 else
 113         setenv=false
 114 fi

 116 WHICH_SCM=$(/bin/dirname $(whence $0))/which_scm
 117 if [[ ! -x $WHICH_SCM ]]; then
 118         WHICH_SCM=which_scm
 119 fi

 121 #
 122 # No workspace/repository path was given, so try and detect one from our
 123 # current directory we're in
 124 #
 125 if [[ $# -lt 1 ]]; then
 126         if env CODEMGR_WS="" $WHICH_SCM | read SCM_MODE tmpwsname && \
 127             [[ $SCM_MODE != unknown ]]; then
 128                 echo "Defaulting to $SCM_MODE repository $tmpwsname"
 129         else
 130                 echo "usage: ws [-e] [workspace_name]" >&2
 131                 if $setenv; then
 132                         cleanup_env
 133                         return 1
 134                 else
 135                         exit 1
 136                 fi
 137         fi
 138 else
 139         #
 140         # A workspace/repository path was passed in, grab it and pop
 141         # it off the stack
 142         #
 143         tmpwsname=$1
 144         shift
 145 fi

 147 #
 148 #       This variable displays the nested activations of workspaces.
 149 #       This is done here to get the exact name the user entered.
 150 #
 151 WS_STACK="$tmpwsname $WS_STACK"; export WS_STACK

 153 #
 154 # Set the workspace name and unset tmpwsname (as we reuse it later)
 155 #
 156 wsname=`echo $tmpwsname|fmtwsname`
 157 unset tmpwsname

 159 #
 160 # Checking for CODEMGR_WSPATH
 161 #
 162 if [[ -n ${CODEMGR_WSPATH} && ( ! -d $wsname ) && \
 163      ( `expr "$wsname" : "\/"` = "0" ) ]]
 164 then
```



```
 165         ofs=$IFS
 166         IFS=":  "
 167         for i in $CODEMGR_WSPATH
 168         do
 169                 if [[ -d ${i}/${wsname} ]]; then
 170                         wsname=${i}/${wsname}
 171                         break
 172                 fi
 173         done
 174         IFS=$ofs
 175 fi

 177 #
 178 # to translate it to an absolute pathname.  We need an
 179 # absolute pathname in order to set CODEMGR_WS.
 180 #
 181 if [[ `expr "$wsname" : "\/"` = "0" ]]
 182 then
 183         pwd=`pwd`
 184         wsname="$pwd/$wsname"
 185 fi

 187 #
 188 #       Check to see if this is a valid workspace
 189 #
 190 if [[ ! -d $wsname ]]; then
 191         echo "$wsname . . . no such directory" >&2
 192         if $setenv; then
 193                 cleanup_env
 194                 return 1
 195         else
 196                 exit 1
 197         fi
 198 fi

 200 #
 201 # This catches the case of a passed in workspace path
 202 # Check which type of SCM is in use by $wsname.
 203 #
 204 (cd $wsname && env CODEMGR_WS="" $WHICH_SCM) | read SCM_MODE tmpwsname
 205 if [[ $? != 0 || "$SCM_MODE" == unknown ]]; then
 206         echo "Error: Unable to detect a supported SCM repository in $wsname"
 207         if $setenv; then
 208                 cleanup_env
 209                 return 1
 210         else
 211                 exit 1
 212         fi
 213 fi

 215 wsname=$tmpwsname
 216 CODEMGR_WS=$wsname ; export CODEMGR_WS
 217 SRC=$wsname/usr/src; export SRC
 218 TSRC=$wsname/usr/ontest; export TSRC

 220 if [[ "$SCM_MODE" = "teamware" && -d ${wsname}/Codemgr_wsdata ]]; then
 221         CM_DATA="Codemgr_wsdata"
 222         wsosdir=$CODEMGR_WS/$CM_DATA/sunos
 223         protofile=$wsosdir/protodefs
 224 elif [[ "$SCM_MODE" = "mercurial" && -d ${wsname}/.hg ]]; then
 225         CM_DATA=".hg"
 226         wsosdir=$CODEMGR_WS/$CM_DATA
 227         protofile=$wsosdir/org.opensolaris.protodefs
 228 elif [[ "$SCM_MODE" = "git" && -d ${wsname}/.git ]]; then
 229         CM_DATA=".git"
 230         wsosdir=$CODEMGR_WS/$CM_DATA
```

```
231         protofile=$wsosdir/org.opensolaris.protodefs
232 else
233         echo "$wsname is not a supported workspace; type is $SCM_MODE" >&2
234         if $setenv; then
235                 cleanup_env
236                 return 1
237         else
238                 exit 1
239         fi
240 fi

242 MACH=`uname -p`

244 if [[ ! -f $protofile ]]; then
245         if [[ ! -w $CODEMGR_WS/$CM_DATA ]]; then
246                 #
247                 # The workspace doesn't have a protodefs file and I am
248                 # unable to create one.  Tell user and use /tmp instead.
249                 #
250                 echo "Unable to create the proto defaults file ($protofile)."

252                 # Just make one in /tmp
253                 wsosdir=/tmp
254                 protofile=$wsosdir/protodefs
255         fi

257         if [[ ! -d $wsosdir ]]; then
258                 mkdir $wsosdir
259         fi

261         cat << PROTOFILE_EoF > $protofile
262 #!/bin/sh
263 #
264 #       Set default proto areas for this workspace
265 #       NOTE: This file was initially automatically generated.
266 #
267 #       Feel free to edit this file.  If this file is removed
268 #       it will be rebuilt containing default values.
269 #
270 #       The variable CODEMGR_WS is available to this script.
271 #
272 #       PROTO1 is the first proto area searched and is typically set
273 #       to a proto area associated with the workspace.  The ROOT
274 #       environment variable is set to the same as PROTO1.  If you
275 #       will be doing make installs this proto area needs to be writable.
276 #
277 #       PROTO2 and PROTO3 are set to proto areas to search before the
278 #       search proceeds to the local machine or the proto area specified by
279 #       TERMPROTO.
280 #
281 #       TERMPROTO (if specified) is the last place searched.  If
282 #       TERMPROTO is not specified the search will end at the local
283 #       machine.
284 #

286 PROTO1=\$CODEMGR_WS/proto
287 PROTOFILE_EoF
288
289         if [[ "$SCM_MODE" = "teamware" ]]; then
290                 cat << PROTOFILE_EoF >> $protofile
291 if [[ -f "\$CODEMGR_WS/Codemgr_wsdata/parent" ]]; then
292    #
293    # If this workspace has an codemgr parent then set PROTO2 to
294    # point to the parents proto space.
295    #
296    parent=\`workspace parent \$CODEMGR_WS\`
```



```
297    if [[ -n \$parent ]]; then
298            PROTO2=\$parent/proto
299    fi
300 fi
301 PROTOFILE_EoF
302         elif [[ "$SCM_MODE" = "mercurial" ]]; then
303                 cat << PROTOFILE_EoF >> $protofile
304 parent=\`(cd \$CODEMGR_WS && hg path default 2>/dev/null)\`
305 if [[ \$? -eq 0 && -n \$parent ]]; then
306    [[ -n \$(check_proto \$parent/proto) ]] && PROTO2=\$parent/proto
307 fi
308 PROTOFILE_EoF
309         fi
310 fi

312 . $protofile

314 # This means you don't have to type make -e all of the time

316 MAKEFLAGS=e; export MAKEFLAGS

318 #
319 #       Set up the environment variables
320 #
321 ROOT=/proto/root_${MACH}        # default

323 ENVCPPFLAGS1=
324 ENVCPPFLAGS2=
325 ENVCPPFLAGS3=
326 ENVCPPFLAGS4=
327 ENVLDLIBS1=
328 ENVLDLIBS2=
329 ENVLDLIBS3=

331 PROTO1=`check_proto $PROTO1`
332 if [[ -n "$PROTO1" ]]; then     # first proto area specifed
333         ROOT=$PROTO1
334         ENVCPPFLAGS1=-I$ROOT/usr/include
335         export ENVCPPFLAGS1
336         ENVLDLIBS1="-L$ROOT/lib -L$ROOT/usr/lib"
337         export ENVLDLIBS1

339         PROTO2=`check_proto $PROTO2`
340         if [[ -n "$PROTO2" ]]; then     # second proto area specifed
341                 ENVCPPFLAGS2=-I$PROTO2/usr/include
342                 export ENVCPPFLAGS2
343                 ENVLDLIBS2="-L$PROTO2/lib -L$PROTO2/usr/lib"
344                 export ENVLDLIBS2

346                 PROTO3=`check_proto $PROTO3`
347                 if [[ -n "$PROTO3" ]]; then     # third proto area specifed
348                         ENVCPPFLAGS3=-I$PROTO3/usr/include
349                         export ENVCPPFLAGS3
350                         ENVLDLIBS3="-L$PROTO3/lib -L$PROTO3/usr/lib"
351                         export ENVLDLIBS3
352                 fi
353         fi
354 fi

356 export ROOT

358 if [[ -n "$TERMPROTO" ]]; then  # fallback area specifed
359         TERMPROTO=`check_proto $TERMPROTO`
360         ENVCPPFLAGS4="-Y I,$TERMPROTO/usr/include"
361         export ENVCPPFLAGS4
362         ENVLDLIBS3="$ENVLDLIBS3 -Y P,$TERMPROTO/lib:$TERMPROTO/usr/lib"
```

```
363         export ENVLDLIBS3
364 fi

366 osbld_flag=0

368 if [[ ! -v CLOSED_IS_PRESENT ]]; then
369         if [[ -d $SRC/../closed ]]; then
370                 export CLOSED_IS_PRESENT="yes"
371         else
372                 export CLOSED_IS_PRESENT="no"
373         fi
374 fi

368 if [[ -z "$ONBLD_DIR" ]]; then
369         ONBLD_DIR=$(/bin/dirname $(whence $0))
370 fi

372 if ! echo ":$PATH:" | grep ":${ONBLD_DIR}:" > /dev/null; then
373         PATH="${ONBLD_DIR}:${ONBLD_DIR}/${MACH}:${PATH}"
374         osbld_flag=1
375 fi

377 export PATH

379 if [[ -n "$PROTO2" ]]; then
380    # This should point to the parent's proto
381    PARENT_ROOT=$PROTO2
382    export PARENT_ROOT
383 else
384    # Clear it in case it's already in the env.
385    PARENT_ROOT=
386 fi
387 export ONBLD_DIR
388 export MACH

390 os_rev=`uname -r`
391 os_name=`uname -s`

393 if [[ $os_name != "SunOS" || `expr $os_rev : "5\."` != "2" ]]; then
394    #
395    # This is not a SunOS 5.x machine - something is wrong
396    #
397    echo "***WARNING: this script is meant to be run on SunOS 5.x."
398    echo "            This machine appears to be running: $os_name $os_rev"
399 fi

401 echo ""
402 echo "Workspace                    : $wsname"
403 if [[ -n "$parent" ]]; then
404    echo "Workspace Parent             : $parent"
405 fi
406 echo "Proto area (\$ROOT)          : $ROOT"
407 if [[ -n "$PARENT_ROOT" ]]; then
408    echo "Parent proto area (\$PARENT_ROOT) : $PARENT_ROOT"
409 fi
410 echo "Root of source (\$SRC)       : $SRC"
411 echo "Root of test source (\$TSRC) : $TSRC"
412 if [[ $osbld_flag = "1" ]]; then
413    echo "Prepended to PATH            : $ONBLD_DIR"
414 fi
415 echo "Current directory (\$PWD)    : $wsname"
416 echo ""

418 cd $wsname

420 if $setenv; then
```



```
421         cleanup_env
422 else
423         exec ${SHELL:-sh} "$@"
424 fi
```

```
**********************************************************
    6649 Wed Oct 16 17:41:08 2013
new/usr/src/uts/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright 2012 Nexenta Systems, Inc. All rights reserved.
#
# include global definitions
include ../Makefile.master

#
# List of architectures to build as part of the standard build.
#
# Some of these architectures are built in parallel (see i386_PARALLEL and
# sparc_PARALLEL). This requires building some parts first before parallel build
# can start. Platform make files know what should be built as a prerequisite for
# the parallel build to work. The i386_PREREQ and sparc_PREREQ variables tell
# which platform directory to enter to start making prerequisite dependencies.
#
sparc_ARCHITECTURES = sun4v sun4u sparc

i386_ARCHITECTURES = i86pc i86xpv intel

#
# For i386 all architectures can be compiled in parallel.
#
# intel/Makefile knows how to build prerequisites needed for parallel build.
#
i386_PREREQ = intel
i386_PARALLEL = $(i386_ARCHITECTURES)

#
# For sparc all architectures can be compiled in parallel.
#
# sun4/Makefile knows how to build prerequisites needed for parallel build.
# can start.
#
sparc_PREREQ = sun4
sparc_PARALLEL = $(sparc_ARCHITECTURES)

#
```



```
# Platforms defined in $(MACH)_PARALLEL are built in parallel. DUMMY is placed
# at the end in case $(MACH)_PARALLEL is empty to prevent everything going in
# parallel.
#
.PARALLEL: $($(MACH)_PARALLEL) DUMMY

#
# For build prerequisites we use a special target which is constructed by adding
# '.prereq' suffix to the $(MACH)_PREREQ.
#
PREREQ_TARGET = $($(MACH)_PREREQ:%=%.prereq)


def             :=      TARGET= def
all             :=      TARGET= all
install         :=      TARGET= install
install_h       :=      TARGET= install_h
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
clobber_h       :=      TARGET= clobber
lint            :=      TARGET= lint
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check
modlist         :=      TARGET= modlist
modlist         :=      NO_STATE= -K $$MODSTATE$$$$

.KEEP_STATE:

def all lint: all_h $(PMTMO_FILE) $($(MACH)_ARCHITECTURES)

install: all_h install_dirs $(PMTMO_FILE) $($(MACH)_ARCHITECTURES)

install_dirs:
        @cd ..; pwd; $(MAKE) rootdirs
        @pwd

#
# Rule to build prerequisites. The left part of the pattern will match
# PREREQ_TARGET.
#
# The location of the Makefile is determined by strippinng '.prereq' suffix from
# the target name. We add '.prereq' suffix to the target passed to the child
# Makefile so that it can distinguish prerequisite build from the regular one.
#
#
%.prereq:
        @cd $(@:%.prereq=%); pwd; $(MAKE) $(NO_STATE) $(TARGET).prereq

#
# Rule to build architecture files. Build all required prerequisites and then
# build the rest (potentially in parallel).
#
$($(MACH)_ARCHITECTURES): $(PREREQ_TARGET) FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

$(PMTMO_FILE) pmtmo_file: $(PATCH_MAKEUP_TABLE)
        @if [ -z "$(PATCH_MAKEUP_TABLE)" ] ; then \
                echo 'ERROR: $$(PATCH_MAKEUP_TABLE) not set' \
                    'in environment' >&2 ; \
                exit 1 ; \
        fi
        RELEASE="$(RELEASE)" MACH="$(MACH)" \
            $(CTFCVTPTBL) -o $(PMTMO_FILE) $(PATCH_MAKEUP_TABLE)

#
# The following is the list of directories which contain Makefiles with
```

```
125 # targets to install header file. The machine independent headers are
126 # installed by invoking the Makefile in the directory containing the
127 # header files. Machine and architecture dependent headers are installed
128 # by invoking the main makefile for that architecture/machine which,
129 # in turn, is responsible for invoking the Makefiles which install headers.
130 # It is done this way so as not to assume that all of the header files in
131 # the architecture/machine dependent subdirectories are in completely
132 # isomorphic locations.
133 #
134 COMMON_HDRDIRS= common/avs \
135                 common/c2 \
136                 common/des \
137                 common/fs \
138                 common/gssapi \
139                 common/idmap \
140                 common/klm \
141                 common/inet \
142                 common/inet/ipf/netinet \
143                 common/inet/kssl \
144                 common/inet/nca \
145                 common/inet/sockmods/netpacket \
146                 common/io/bpf/net \
147                 common/io/fibre-channel/fca/qlc \
148                 common/io/lvm/md \
149                 common/ipp \
150                 common/net \
151                 common/netinet \
152                 common/nfs \
153                 common/pcmcia/sys \
154                 common/rpc \
155                 common/rpcsvc \
156                 common/sharefs \
157                 common/smb \
158                 common/smbsrv \
159                 common/sys \
160                 common/vm


163 # These aren't the only headers in closed.  But the other directories
164 # are simple enough that they can be driven from the src tree.
165 $(CLOSED_BUILD)COMMON_HDRDIRS += $(CLOSED)/uts/common/sys

163 #
164 # Subset of COMMON_HDRDIRS in which at least one header is generated
165 # at runtime (e.g., rpcgen), and in which "make clean" should run.
166 # Other directories should be included here, but do not yet have the
167 # necessary Makefile support (make clean).  See 6414855.
168 #
169 DYNHDRDIRS =    common/avs \
170                 common/gssapi \
171                 common/idmap \
172                 common/io/fibre-channel/fca/qlc \
173                 common/io/lvm/md \
174                 common/klm \
175                 common/rpc \
176                 common/rpcsvc \
177                 common/sys

179 sparc_HDRDIRS= sun/sys
180 i386_HDRDIRS= i86pc/vm i86xpv/vm

182 HDRDIRS= $(COMMON_HDRDIRS) $($(MACH)_HDRDIRS)
183 install_h check: $(HDRDIRS) $($(MACH)_ARCHITECTURES)

185 $(HDRDIRS): FRC
186         @cd $@; pwd; $(MAKE) $(TARGET)
```



```
188 # ensures that headers made by rpcgen and others are available in uts source
189 # for kernel builds to reference without building install_h
190 #
191 all_h: FRC
192         @cd common/sys; pwd; $(MAKE) $@
193         @cd common/rpc; pwd; $(MAKE) $@
194         @cd common/rpcsvc; pwd; $(MAKE) $@
195         @cd common/gssapi; pwd; $(MAKE) $@
196         @cd common/idmap; pwd; $(MAKE) $@
197         @cd common/klm; pwd; $(MAKE) $@

199 clean clobber: $($(MACH)_ARCHITECTURES) $(DYNHDRDIRS)
200         @if [ '$(PATCH_BUILD)' != '#' ] ; then \
201                 echo $(RM) $(PMTMO_FILE) ; \
202                 $(RM) $(PMTMO_FILE) ; \
203         fi

205 # testing convenience
206 clobber_h: $(DYNHDRDIRS)

208 clean.lint modlist: $($(MACH)_ARCHITECTURES)

210 #
211 # Cross-reference customization: build a cross-reference over all of
212 # the supported architectures.  Although there's no correct way to set
213 # the include path (since we don't know what architecture is the one
214 # the user will be interested in), it's historically been set to
215 # mirror the $(XRDIRS) list, and that works kinda sorta okay.
216 #
217 XRDIRS = $(sparc_ARCHITECTURES) $(i386_ARCHITECTURES) sun4 sfmmu        \
221 # We need to manually prune usr/closed/uts/{i86xpv|sfmmu|i86pc} since
222 # none of them exist.
223 #
224 SHARED_XRDIRS = $(sparc_ARCHITECTURES) $(i386_ARCHITECTURES) sun4 sfmmu \
218         sun common
226 CLOSED_XRDIRS = $(SHARED_XRDIRS:%=% ../../closed/uts/%)
227 XRDIRS = $(SHARED_XRDIRS)
228 CLOSED_XRDIRS_XEN = $(CLOSED_XRDIRS:../../closed/uts/i86xpv=)
229 CLOSED_XRDIRS_1 = $(CLOSED_XRDIRS_XEN:../../closed/uts/i86pc=)
230 $(CLOSED_BUILD)XRDIRS = $(CLOSED_XRDIRS_1:../../closed/uts/sfmmu=)

220 XRINCDIRS = $(XRDIRS)

222 cscope.out tags: FRC
223         $(XREF) -x $@

225 FRC:
```

```
**********************************************************
   22676 Wed Oct 16 17:41:08 2013
new/usr/src/uts/Makefile.uts
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
# Copyright (c) 2011 by Delphix. All rights reserved.
# Copyright (c) 2013 Andrew Stormont.  All rights reserved.
#

#
#       This Makefile contains the common targets and definitions for
#       all kernels. It is to be included in the Makefiles for specific
#       implementation architectures and processor architecture dependent
#       modules: i.e.: all driving kernel Makefiles.
#
#       Include global definitions:
#
include $(SRC)/Makefile.master

#
#       No text domain in the kernel.
#
DTEXTDOM =

#
#       Keep references to $(SRC)/common relative.
COMMONBASE=     $(UTSBASE)/../common

#
#       Setup build-specific vars
#       To add a build type:
#               add name to ALL_BUILDS32 & ALL_BUILDS64
#               set CLASS_name and OBJ_DIR_name
#               add targets to Makefile.targ
#

#
#       DEF_BUILDS is for def, lint, sischeck, and install
#       ALL_BUILDS is for everything else (all, clean, ...)
```



```
#
# The NOT_RELEASE_BUILD noise is to maintain compatibility with the
# gatekeeper's nightly build script.
#
DEF_BUILDS32                            = obj32
DEF_BUILDS64                            = obj64
DEF_BUILDSONLY64                        = obj64
$(NOT_RELEASE_BUILD)DEF_BUILDS32        = debug32
$(NOT_RELEASE_BUILD)DEF_BUILDS64        = debug64
$(NOT_RELEASE_BUILD)DEF_BUILDSONLY64    = debug64
ALL_BUILDS32                            = obj32 debug32
ALL_BUILDS64                            = obj64 debug64
ALL_BUILDSONLY64                        = obj64 debug64

#
#       For modules in 64b dirs that aren't built 64b
#       or modules in 64b dirs that aren't built 32b we
#       need to create empty modlintlib files so global lint works
#
LINT32_BUILDS   = debug32
LINT64_BUILDS   = debug64

#
#       Build class (32b or 64b)
#
CLASS_OBJ32     = 32
CLASS_DBG32     = 32
CLASS_OBJ64     = 64
CLASS_DBG64     = 64
CLASS           = $(CLASS_$(BUILD_TYPE))

#
#       Build subdirectory
#
OBJS_DIR_OBJ32  = obj32
OBJS_DIR_DBG32  = debug32
OBJS_DIR_OBJ64  = obj64
OBJS_DIR_DBG64  = debug64
OBJS_DIR        = $(OBJS_DIR_$(BUILD_TYPE))

#
#       Create defaults so empty rules don't
#       confuse make
#
CLASS_          = 32
OBJS_DIR_       = debug32

#
#       Build tools
#
CC_sparc_32     = $(sparc_CC)
CC_sparc_64     = $(sparcv9_CC)

CC_i386_32      = $(i386_CC)
CC_i386_64      = $(amd64_CC)
CC_amd64_64     = $(amd64_CC)

CC              = $(CC_$(MACH)_$(CLASS))

AS_sparc_32     = $(sparc_AS)
AS_sparc_64     = $(sparcv9_AS)

AS_i386_32      = $(i386_AS)
AS_i386_64      = $(amd64_AS)
AS_amd64_64     = $(amd64_AS)
```

```
125 AS              = $(AS_$(MACH)_$(CLASS))

127 LD_sparc_32     = $(sparc_LD)
128 LD_sparc_64     = $(sparcv9_LD)

130 LD_i386_32      = $(i386_LD)
131 LD_i386_64      = $(amd64_LD)
132 LD_amd64_64     = $(amd64_LD)

134 LD              = $(LD_$(MACH)_$(CLASS))

136 LINT_sparc_32   = $(sparc_LINT)
137 LINT_sparc_64   = $(sparcv9_LINT)

139 LINT_i386_32    = $(i386_LINT)
140 LINT_i386_64    = $(amd64_LINT)
141 LINT_amd64_64   = $(amd64_LINT)

143 LINT            = $(LINT_$(MACH)_$(CLASS))

145 MODEL_32        = ilp32
146 MODEL_64        = lp64
147 MODEL           = $(MODEL_$(CLASS))

149 #
150 #       Build rules for linting the kernel.
151 #
152 LHEAD = $(ECHO) "\n$@";

154 # Note: egrep returns "failure" if there are no matches, which is
155 # exactly the opposite of what we need.
156 LGREP.2 =       if egrep -v ' (_init|_fini|_info) '; then false; else true; fi

158 LTAIL =

160 LINT.c =        $(LINT) -c -dirout=$(LINTS_DIR) $(LINTFLAGS) $(LINT_DEFS) $(CPPF

162 # Please do not add new erroff directives here.  If you need to disable
163 # lint warnings in your module for things that cannot be fixed in any
164 # reasonable manner, please augment LINTTAGS in your module Makefile
165 # instead.
166 LINTTAGS        = -erroff=E_INCONS_ARG_DECL2
167 LINTTAGS        += -erroff=E_INCONS_VAL_TYPE_DECL2

169 LINTFLAGS_sparc_32      = $(LINTCCMODE) -nsxmuF -errtags=yes
170 LINTFLAGS_sparc_64      = $(LINTFLAGS_sparc_32) -m64
171 LINTFLAGS_i386_32       = $(LINTCCMODE) -nsxmuF -errtags=yes
172 LINTFLAGS_i386_64       = $(LINTFLAGS_i386_32) -m64

174 LINTFLAGS       = $(LINTFLAGS_$(MACH)_$(CLASS)) $(LINTTAGS)
175 LINTFLAGS       += $(C99LMODE)

177 #
178 #       Override this variable to modify the name of the lint target.
179 #
180 LINT_MODULE=    $(MODULE)

182 #
183 #       Build the compile/assemble lines:
184 #
185 EXTRA_OPTIONS           =
186 AS_DEFS                 = -D_ASM -D__STDC__=0

188 ALWAYS_DEFS_32          = -D_KERNEL -D_SYSCALL32 -D_DDI_STRICT
189 ALWAYS_DEFS_64          = -D_KERNEL -D_SYSCALL32 -D_SYSCALL32_IMPL -D_ELF64 \
190                         -D_DDI_STRICT
```



```
191 #
192 # XX64  This should be defined by the compiler!
193 #
194 ALWAYS_DEFS_64          += -Dsun -D__sun -D__SVR4
195 ALWAYS_DEFS             = $(ALWAYS_DEFS_$(CLASS))

197 #
198 #       CPPFLAGS is deliberatly set with a "=" and not a "+=".  For the kernel
199 #       the header include path should not look for header files outside of
200 #       the kernel code.  This "=" removes the search path built in
201 #       Makefile.master inside CPPFLAGS.  Ditto for AS_CPPFLAGS.
202 #
203 CPPFLAGS        = $(ALWAYS_DEFS) $(ALL_DEFS) $(CONFIG_DEFS) \
204                     $(INCLUDE_PATH) $(EXTRA_OPTIONS)
205 ASFLAGS         += -P
206 AS_CPPFLAGS     = $(ALWAYS_DEFS) $(ALL_DEFS) $(CONFIG_DEFS) $(AS_DEFS) \
207                     $(AS_INC_PATH) $(EXTRA_OPTIONS)

209 #
210 #       Make it (relatively) easy to share compilation options between
211 #       all kernel implementations.
212 #

214 # Override the default, the kernel is squeaky clean
215 CERRWARN = -errtags=yes -errwarn=%all

217 CERRWARN += -_gcc=-Wno-missing-braces
218 CERRWARN += -_gcc=-Wno-sign-compare
219 CERRWARN += -_gcc=-Wno-unknown-pragmas
220 CERRWARN += -_gcc=-Wno-unused-parameter
221 CERRWARN += -_gcc=-Wno-missing-field-initializers

223 # DEBUG v. -nd make for frequent unused variables, empty conditions, etc. in
224 # -nd builds
225 $(RELEASE_BUILD)CERRWARN += -_gcc=-Wno-unused
226 $(RELEASE_BUILD)CERRWARN += -_gcc=-Wno-empty-body

228 C99MODE = $(C99_ENABLE)

230 CFLAGS_uts              =
231 CFLAGS_uts              += $(STAND_FLAGS_$(CLASS))
232 CFLAGS_uts              += $(CCVERBOSE)
233 CFLAGS_uts              += $(ILDOFF)
234 CFLAGS_uts              += $(XAOPT)
235 CFLAGS_uts              += $(CTF_FLAGS_$(CLASS))
236 CFLAGS_uts              += $(CERRWARN)
237 CFLAGS_uts              += $(CCNOAUTOINLINE)
238 CFLAGS_uts              += $(CGLOBALSTATIC)
239 CFLAGS_uts              += $(EXTRA_CFLAGS)
240 CFLAGS_uts              += $(CSOURCEDEBUGFLAGS)
241 CFLAGS_uts              += $(CUSERFLAGS)

243 #
244 #       Declare that $(OBJECTS) and $(LINTS) can be compiled in parallel.
245 #       The DUMMY target is for those instances where OBJECTS and LINTS
246 #       are empty (to avoid an unconditional .PARALLEL).
247 .PARALLEL:      $(OBJECTS) $(LINTS) DUMMY

249 #
250 #       Expanded dependencies
251 #
252 DEF_DEPS        = $(DEF_BUILDS:%=def.%)
253 ALL_DEPS        = $(ALL_BUILDS:%=all.%)
254 CLEAN_DEPS      = $(ALL_BUILDS:%=clean.%)
255 CLOBBER_DEPS    = $(ALL_BUILDS:%=clobber.%)
256 LINT_DEPS       = $(DEF_BUILDS:%=lint.%)
```

```
257 MODLINTLIB_DEPS = $(DEF_BUILDS:%=modlintlib.%)
258 MODLIST_DEPS    = $(DEF_BUILDS:%=modlist.%)
259 CLEAN_LINT_DEPS = $(ALL_BUILDS:%=clean.lint.%)
260 INSTALL_DEPS    = $(DEF_BUILDS:%=install.%)
261 SYM_DEPS        = $(SYM_BUILDS:%=symcheck.%)
262 SISCHECK_DEPS   = $(DEF_BUILDS:%=sischeck.%)
263 SISCLEAN_DEPS   = $(ALL_BUILDS:%=sisclean.%)

265 #
266 #       Default module name
267 #
268 BINARY          = $(OBJS_DIR)/$(MODULE)

270 #
271 #       Default cleanup definitions
272 #
273 CLEANLINTFILES  = $(LINTS) $(MOD_LINT_LIB)
274 CLEANFILES      = $(OBJECTS) $(CLEANLINTFILES)
275 CLOBBERFILES    = $(BINARY) $(CLEANFILES)

277 #
278 #       Installation constants:
279 #
280 #               FILEMODE is the mode given to the kernel modules
281 #               CFILEMODE is the mode given to the '.conf' files
282 #
283 FILEMODE         = 755
284 DIRMODE          = 755
285 CFILEMODE        = 644

287 #
288 #       Special Installation Macros for the installation of '.conf' files.
289 #
290 #       These are unique because they are not installed from the current
291 #       working directory.
292 #
293 # Sigh.  Apparently at some time in the past there was a confusion on
294 # whether the name is SRC_CONFFILE or SRC_CONFILE.  Consistency with the
295 # other names would indicate SRC_CONFFILE, but the voting is >180 Makefiles
296 # with SRC_CONFILE and about 11 with SRC_CONFFILE.  Software development
297 # isn't a popularity contest, though, and so my inclination is to define
298 # both names for now and incrementally convert to SRC_CONFFILE to be consistent
299 # with the other names.
300 #
301 CONFFILE                = $(MODULE).conf
302 SRC_CONFFILE            = $(CONF_SRCDIR)/$(CONFFILE)
303 SRC_CONFILE             = $(SRC_CONFFILE)
304 ROOT_CONFFILE_32        = $(ROOTMODULE).conf
305 ROOT_CONFFILE_64        = $(ROOTMODULE:%/$(SUBDIR64)/$(MODULE)=%/$(MODULE)).conf
306 ROOT_CONFFILE           = $(ROOT_CONFFILE_$(CLASS))


309 INS.conffile= \
310         $(RM) $@; $(INS) -s -m $(CFILEMODE) -f $(@D) $(SRC_CONFFILE)

312 #
313 # The CTF merge of child kernel modules is performed against one of the genunix
314 # modules.  For Intel builds, all modules will be used with a single genunix:
315 # the one built in intel/genunix.  For SPARC builds, a given
316 # module may be
317 # used with one of a number of genunix files, depending on what platform the
318 # module is deployed on.  We merge against the sun4u genunix to optimize for
319 # the common case.  We also merge against the ip driver since networking is
320 # typically loaded and types defined therein are shared between many modules.
321 #
322 CTFMERGE_GUDIR_sparc    = sun4u
```



```
323 CTFMERGE_GUDIR_i386     = intel
324 CTFMERGE_GUDIR          = $(CTFMERGE_GUDIR_$(MACH))

326 CTFMERGE_GENUNIX        = \
327         $(UTSBASE)/$(CTFMERGE_GUDIR)/genunix/$(OBJS_DIR)/genunix

329 #
330 # Used to uniquify a non-genunix module against genunix.  If used in patch
331 # mode (PATCH_BUILD != "#"), the patch ID corresponding to the module being
332 # built will be used as the label.  If no ID is available, or if patch mode
333 # is not being used, the value of $VERSION will be used.
334 #
335 # For the ease of developers dropping modules onto possibly unrelated systems,
336 # you can set NO_GENUNIX_UNIQUIFY= in the environment to skip uniquifying
337 # against genunix.
338 #
339 NO_GENUNIX_UNIQUIFY=$(POUND_SIGN)
340 SKIP_GENUNIX_UNIQUIFY=no
341 $(NO_GENUNIX_UNIQUIFY)SKIP_GENUNIX_UNIQUIFY=yes

343 CTFMERGE_UNIQUIFY_AGAINST_GENUNIX       = \
344         @label="-L VERSION" ; \
345         uniq= ; \
346         if [ -z "$(PATCH_BUILD)" ] ; then \
347                 uniq="-D BASE" ; \
348                 set -- '$(CTFFINDMOD) -n -r -t $(PMTMO_FILE) $@' ; \
349                 if [ "X$$1" != "X-" ] ; then \
350                         label="-l $$1" ; \
351                         if [ "$$2" != "fcs" ] ; then \
352                                 uniq="-D $$2" ; \
353                         fi ; \
354                 fi ; \
355         fi ; \
356         if [ "$(SKIP_GENUNIX_UNIQUIFY)" = "yes" ]; then \
357                 uniq= ; \
358         else \
359                 uniq="-d $(CTFMERGE_GENUNIX) $$uniq" ; \
360         fi ; \
361         cmd="$(CTFMERGE) $(CTFMRGFLAGS) $$label $$uniq" ; \
362         cmd="$$cmd -o $@ $(OBJECTS) $(CTFEXTRAOBJS)" ; \
363         echo $$cmd ; \
364         $$cmd

366 #
367 # Used to merge the genunix module.  genunix has special requirements in
368 # patch mode.  In particular, it needs to be able to find the genunix used
369 # in the previous version of the KU patch (or the FCS version of genunix in
370 # the case of KU 1).
371 #
372 CTFMERGE_GENUNIX_MERGE          = \
373         @if [ -z "$(PATCH_BUILD)" ] ; then \
374                 set -- '$(CTFFINDMOD) -b $(OBJS_DIR) -o patch,lastgu -n -r \
375                     -t $(PMTMO_FILE) $(GENUNIX) || true' '' ; \
376                 msg= ; \
377                 if [ $$$(POUND_SIGN) -eq 1 ] ; \
378                     then msg="Error in $(CTFFINDMOD)" ; \
379                 elif [ "X$$1" = "X-" ] ; then msg="Did not get label" ; \
380                 elif [ "X$$2" = "X-" ] ; then msg="Did not get withfile" ; \
381                 fi ; \
382                 if [ -n "$$msg" ] ; then \
383                         echo "make ctf: $$msg - removing $(GENUNIX)" ; \
384                         $(RM) $(GENUNIX) ; \
385                         exit 1 ; \
386                 fi ; \
387                 label="-l $$1" ; \
388                 with="-w $$2" ; \
```

```
389         else \
390                 label="-L VERSION" ; \
391         fi ; \
392         cmd="$(CTFMERGE) $(CTFMRGFLAGS) $$label $$with -o $@" ; \
393         echo $$cmd "$(OBJECTS) $(CTFEXTRAOBJS) $(IPCTF_TARGET)"; \
394         $$cmd $(OBJECTS) $(CTFEXTRAOBJS) $(IPCTF_TARGET)

396 #
397 # We ctfmerge the ip objects into genunix to maximize the number of common types
398 # found there, thus maximizing the effectiveness of uniquification.  We don't
399 # want the genunix build to have to know about the individual ip objects, so we
400 # put them in an archive.  The genunix ctfmerge then includes this archive.
401 #
402 IPCTF           = $(IPDRV_DIR)/$(OBJS_DIR)/ipctf.a

404 #
405 # Rule for building fake shared libraries used for symbol resolution
406 # when building other modules.  -znoreloc is needed here to avoid
407 # tripping over code that isn't really suitable for shared libraries.
408 #
409 BUILD.SO                = \
410         $(LD) -o $@ $(GSHARED) $(ZNORELOC) -h $(SONAME)

412 #
413 # SONAME defaults for common fake shared libraries.
414 #
415 $(LIBGEN)               := SONAME = $(MODULE)
416 $(PLATLIB)              := SONAME = misc/platmod
417 $(CPULIB)               := SONAME = 'cpu/$$CPU'
418 $(DTRACESTUBS)          := SONAME = dtracestubs

420 #
421 #       Installation directories
422 #

424 #
425 #       For now, 64b modules install into a subdirectory
426 #       of their 32b brethren.
427 #
428 SUBDIR64_sparc          = sparcv9
429 SUBDIR64_i386           = amd64
430 SUBDIR64                = $(SUBDIR64_$(MACH))

432 ROOT_MOD_DIR            = $(ROOT)/kernel

434 ROOT_KERN_DIR_32        = $(ROOT_MOD_DIR)
435 ROOT_BRAND_DIR_32       = $(ROOT_MOD_DIR)/brand
436 ROOT_DRV_DIR_32         = $(ROOT_MOD_DIR)/drv
437 ROOT_DTRACE_DIR_32      = $(ROOT_MOD_DIR)/dtrace
438 ROOT_EXEC_DIR_32        = $(ROOT_MOD_DIR)/exec
439 ROOT_FS_DIR_32          = $(ROOT_MOD_DIR)/fs
440 ROOT_SCHED_DIR_32       = $(ROOT_MOD_DIR)/sched
441 ROOT_SOCK_DIR_32        = $(ROOT_MOD_DIR)/socketmod
442 ROOT_STRMOD_DIR_32      = $(ROOT_MOD_DIR)/strmod
443 ROOT_IPP_DIR_32         = $(ROOT_MOD_DIR)/ipp
444 ROOT_SYS_DIR_32         = $(ROOT_MOD_DIR)/sys
445 ROOT_MISC_DIR_32        = $(ROOT_MOD_DIR)/misc
446 ROOT_KGSS_DIR_32        = $(ROOT_MOD_DIR)/misc/kgss
447 ROOT_SCSI_VHCI_DIR_32   = $(ROOT_MOD_DIR)/misc/scsi_vhci
448 ROOT_PMCS_FW_DIR_32     = $(ROOT_MOD_DIR)/misc/pmcs
449 ROOT_QLC_FW_DIR_32      = $(ROOT_MOD_DIR)/misc/qlc
450 ROOT_EMLXS_FW_DIR_32    = $(ROOT_MOD_DIR)/misc/emlxs
451 ROOT_NLMISC_DIR_32      = $(ROOT_MOD_DIR)/misc
452 ROOT_MACH_DIR_32        = $(ROOT_MOD_DIR)/mach
453 ROOT_CPU_DIR_32         = $(ROOT_MOD_DIR)/cpu
454 ROOT_TOD_DIR_32         = $(ROOT_MOD_DIR)/tod
```



```
455 ROOT_FONT_DIR_32        = $(ROOT_MOD_DIR)/fonts
456 ROOT_DACF_DIR_32        = $(ROOT_MOD_DIR)/dacf
457 ROOT_CRYPTO_DIR_32      = $(ROOT_MOD_DIR)/crypto
458 ROOT_MAC_DIR_32         = $(ROOT_MOD_DIR)/mac
459 ROOT_KICONV_DIR_32      = $(ROOT_MOD_DIR)/kiconv

461 ROOT_KERN_DIR_64        = $(ROOT_MOD_DIR)/$(SUBDIR64)
462 ROOT_BRAND_DIR_64       = $(ROOT_MOD_DIR)/brand/$(SUBDIR64)
463 ROOT_DRV_DIR_64         = $(ROOT_MOD_DIR)/drv/$(SUBDIR64)
464 ROOT_DTRACE_DIR_64      = $(ROOT_MOD_DIR)/dtrace/$(SUBDIR64)
465 ROOT_EXEC_DIR_64        = $(ROOT_MOD_DIR)/exec/$(SUBDIR64)
466 ROOT_FS_DIR_64          = $(ROOT_MOD_DIR)/fs/$(SUBDIR64)
467 ROOT_SCHED_DIR_64       = $(ROOT_MOD_DIR)/sched/$(SUBDIR64)
468 ROOT_SOCK_DIR_64        = $(ROOT_MOD_DIR)/socketmod/$(SUBDIR64)
469 ROOT_STRMOD_DIR_64      = $(ROOT_MOD_DIR)/strmod/$(SUBDIR64)
470 ROOT_IPP_DIR_64         = $(ROOT_MOD_DIR)/ipp/$(SUBDIR64)
471 ROOT_SYS_DIR_64         = $(ROOT_MOD_DIR)/sys/$(SUBDIR64)
472 ROOT_MISC_DIR_64        = $(ROOT_MOD_DIR)/misc/$(SUBDIR64)
473 ROOT_KGSS_DIR_64        = $(ROOT_MOD_DIR)/misc/kgss/$(SUBDIR64)
474 ROOT_SCSI_VHCI_DIR_64   = $(ROOT_MOD_DIR)/misc/scsi_vhci/$(SUBDIR64)
475 ROOT_PMCS_FW_DIR_64     = $(ROOT_MOD_DIR)/misc/pmcs/$(SUBDIR64)
476 ROOT_QLC_FW_DIR_64      = $(ROOT_MOD_DIR)/misc/qlc/$(SUBDIR64)
477 ROOT_EMLXS_FW_DIR_64    = $(ROOT_MOD_DIR)/misc/emlxs/$(SUBDIR64)
478 ROOT_NLMISC_DIR_64      = $(ROOT_MOD_DIR)/misc/$(SUBDIR64)
479 ROOT_MACH_DIR_64        = $(ROOT_MOD_DIR)/mach/$(SUBDIR64)
480 ROOT_CPU_DIR_64         = $(ROOT_MOD_DIR)/cpu/$(SUBDIR64)
481 ROOT_TOD_DIR_64         = $(ROOT_MOD_DIR)/tod/$(SUBDIR64)
482 ROOT_FONT_DIR_64        = $(ROOT_MOD_DIR)/fonts/$(SUBDIR64)
483 ROOT_DACF_DIR_64        = $(ROOT_MOD_DIR)/dacf/$(SUBDIR64)
484 ROOT_CRYPTO_DIR_64      = $(ROOT_MOD_DIR)/crypto/$(SUBDIR64)
485 ROOT_MAC_DIR_64         = $(ROOT_MOD_DIR)/mac/$(SUBDIR64)
486 ROOT_KICONV_DIR_64      = $(ROOT_MOD_DIR)/kiconv/$(SUBDIR64)

488 ROOT_KERN_DIR           = $(ROOT_KERN_DIR_$(CLASS))
489 ROOT_BRAND_DIR          = $(ROOT_BRAND_DIR_$(CLASS))
490 ROOT_DRV_DIR            = $(ROOT_DRV_DIR_$(CLASS))
491 ROOT_DTRACE_DIR         = $(ROOT_DTRACE_DIR_$(CLASS))
492 ROOT_EXEC_DIR           = $(ROOT_EXEC_DIR_$(CLASS))
493 ROOT_FS_DIR             = $(ROOT_FS_DIR_$(CLASS))
494 ROOT_SCHED_DIR          = $(ROOT_SCHED_DIR_$(CLASS))
495 ROOT_SOCK_DIR           = $(ROOT_SOCK_DIR_$(CLASS))
496 ROOT_STRMOD_DIR         = $(ROOT_STRMOD_DIR_$(CLASS))
497 ROOT_IPP_DIR            = $(ROOT_IPP_DIR_$(CLASS))
498 ROOT_SYS_DIR            = $(ROOT_SYS_DIR_$(CLASS))
499 ROOT_MISC_DIR           = $(ROOT_MISC_DIR_$(CLASS))
500 ROOT_KGSS_DIR           = $(ROOT_KGSS_DIR_$(CLASS))
501 ROOT_SCSI_VHCI_DIR      = $(ROOT_SCSI_VHCI_DIR_$(CLASS))
502 ROOT_PMCS_FW_DIR        = $(ROOT_PMCS_FW_DIR_$(CLASS))
503 ROOT_QLC_FW_DIR         = $(ROOT_QLC_FW_DIR_$(CLASS))
504 ROOT_EMLXS_FW_DIR       = $(ROOT_EMLXS_FW_DIR_$(CLASS))
505 ROOT_NLMISC_DIR         = $(ROOT_NLMISC_DIR_$(CLASS))
506 ROOT_MACH_DIR           = $(ROOT_MACH_DIR_$(CLASS))
507 ROOT_CPU_DIR            = $(ROOT_CPU_DIR_$(CLASS))
508 ROOT_TOD_DIR            = $(ROOT_TOD_DIR_$(CLASS))
509 ROOT_FONT_DIR           = $(ROOT_FONT_DIR_$(CLASS))
510 ROOT_DACF_DIR           = $(ROOT_DACF_DIR_$(CLASS))
511 ROOT_CRYPTO_DIR         = $(ROOT_CRYPTO_DIR_$(CLASS))
512 ROOT_MAC_DIR            = $(ROOT_MAC_DIR_$(CLASS))
513 ROOT_KICONV_DIR         = $(ROOT_KICONV_DIR_$(CLASS))

515 ROOT_MOD_DIRS_32        = $(ROOT_BRAND_DIR_32) $(ROOT_DRV_DIR_32)
516 ROOT_MOD_DIRS_32        = $(ROOT_BRAND_DIR_32) $(ROOT_DRV_DIR_32)
517 ROOT_MOD_DIRS_32        += $(ROOT_EXEC_DIR_32) $(ROOT_DTRACE_DIR_32)
518 ROOT_MOD_DIRS_32        += $(ROOT_FS_DIR_32) $(ROOT_SCHED_DIR_32)
519 ROOT_MOD_DIRS_32        += $(ROOT_STRMOD_DIR_32) $(ROOT_SYS_DIR_32)
520 ROOT_MOD_DIRS_32        += $(ROOT_IPP_DIR_32) $(ROOT_SOCK_DIR_32)
```

```
ROOT_MOD_DIRS_32         += $(ROOT_MISC_DIR_32) $(ROOT_MACH_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_KGSS_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_SCSI_VHCI_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_PMCS_FW_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_QLC_FW_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_EMLXS_FW_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_CPU_DIR_32) $(ROOT_FONT_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_TOD_DIR_32) $(ROOT_DACF_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_CRYPTO_DIR_32) $(ROOT_MAC_DIR_32)
ROOT_MOD_DIRS_32         += $(ROOT_KICONV_DIR_32)

USR_MOD_DIR              = $(ROOT)/usr/kernel

USR_DRV_DIR_32           = $(USR_MOD_DIR)/drv
USR_EXEC_DIR_32          = $(USR_MOD_DIR)/exec
USR_FS_DIR_32            = $(USR_MOD_DIR)/fs
USR_SCHED_DIR_32         = $(USR_MOD_DIR)/sched
USR_SOCK_DIR_32          = $(USR_MOD_DIR)/socketmod
USR_STRMOD_DIR_32        = $(USR_MOD_DIR)/strmod
USR_SYS_DIR_32           = $(USR_MOD_DIR)/sys
USR_MISC_DIR_32          = $(USR_MOD_DIR)/misc
USR_DACF_DIR_32          = $(USR_MOD_DIR)/dacf
USR_PCBE_DIR_32          = $(USR_MOD_DIR)/pcbe
USR_DTRACE_DIR_32        = $(USR_MOD_DIR)/dtrace
USR_BRAND_DIR_32         = $(USR_MOD_DIR)/brand

USR_DRV_DIR_64           = $(USR_MOD_DIR)/drv/$(SUBDIR64)
USR_EXEC_DIR_64          = $(USR_MOD_DIR)/exec/$(SUBDIR64)
USR_FS_DIR_64            = $(USR_MOD_DIR)/fs/$(SUBDIR64)
USR_SCHED_DIR_64         = $(USR_MOD_DIR)/sched/$(SUBDIR64)
USR_SOCK_DIR_64          = $(USR_MOD_DIR)/socketmod/$(SUBDIR64)
USR_STRMOD_DIR_64        = $(USR_MOD_DIR)/strmod/$(SUBDIR64)
USR_SYS_DIR_64           = $(USR_MOD_DIR)/sys/$(SUBDIR64)
USR_MISC_DIR_64          = $(USR_MOD_DIR)/misc/$(SUBDIR64)
USR_DACF_DIR_64          = $(USR_MOD_DIR)/dacf/$(SUBDIR64)
USR_PCBE_DIR_64          = $(USR_MOD_DIR)/pcbe/$(SUBDIR64)
USR_DTRACE_DIR_64        = $(USR_MOD_DIR)/dtrace/$(SUBDIR64)
USR_BRAND_DIR_64         = $(USR_MOD_DIR)/brand/$(SUBDIR64)

USR_DRV_DIR              = $(USR_DRV_DIR_$(CLASS))
USR_EXEC_DIR             = $(USR_EXEC_DIR_$(CLASS))
USR_FS_DIR               = $(USR_FS_DIR_$(CLASS))
USR_SCHED_DIR            = $(USR_SCHED_DIR_$(CLASS))
USR_SOCK_DIR             = $(USR_SOCK_DIR_$(CLASS))
USR_STRMOD_DIR           = $(USR_STRMOD_DIR_$(CLASS))
USR_SYS_DIR              = $(USR_SYS_DIR_$(CLASS))
USR_MISC_DIR             = $(USR_MISC_DIR_$(CLASS))
USR_DACF_DIR             = $(USR_DACF_DIR_$(CLASS))
USR_PCBE_DIR             = $(USR_PCBE_DIR_$(CLASS))
USR_DTRACE_DIR           = $(USR_DTRACE_DIR_$(CLASS))
USR_BRAND_DIR            = $(USR_BRAND_DIR_$(CLASS))

USR_MOD_DIRS_32          = $(USR_DRV_DIR_32) $(USR_EXEC_DIR_32)
USR_MOD_DIRS_32          += $(USR_FS_DIR_32) $(USR_SCHED_DIR_32)
USR_MOD_DIRS_32          += $(USR_STRMOD_DIR_32) $(USR_SYS_DIR_32)
USR_MOD_DIRS_32          += $(USR_MISC_DIR_32) $(USR_DACF_DIR_32)
USR_MOD_DIRS_32          += $(USR_PCBE_DIR_32)
USR_MOD_DIRS_32          += $(USR_DTRACE_DIR_32) $(USR_BRAND_DIR_32)
USR_MOD_DIRS_32          += $(USR_SOCK_DIR_32)

#
#
#
include $(SRC)/Makefile.psm

#
```



```
#       The "-r" on the remove may be considered temporary, but is required
#       while the replacement of the SUNW,SPARCstation-10,SX directory by
#       a symbolic link is being propagated.
#
# IMPORTANT:: if you change any of these INS.mumble rules, then you MUST also
# change the corresponding override definitions in $CLOSED/Makefile.tonic.
# If you do not do this, then the closedbins build for the OpenSolaris
# community will break. PS, the gatekeepers will be upset too.
#
INS.slink1= $(RM) -r $@; $(SYMLINK) $(PLATFORM) $@
INS.slink2= $(RM) -r $@; $(SYMLINK) ../$(PLATFORM)/$(@F) $@
INS.slink3= $(RM) -r $@; $(SYMLINK) $(IMPLEMENTED_PLATFORM) $@
INS.slink4= $(RM) -r $@; $(SYMLINK) ../$(PLATFORM)/include $@
INS.slink5= $(RM) -r $@; $(SYMLINK) ../$(PLATFORM)/sbin $@
INS.slink6= $(RM) -r $@; $(SYMLINK) ../../$(PLATFORM)/lib/$(MODULE) $@
INS.slink7= $(RM) -r $@; $(SYMLINK) ../../$(PLATFORM)/sbin/$(@F) $@

ROOT_PLAT_LINKS          = $(PLAT_LINKS:%=$(ROOT_PLAT_DIR)/%)
ROOT_PLAT_LINKS_2        = $(PLAT_LINKS_2:%=$(ROOT_PLAT_DIR)/%)
USR_PLAT_LINKS           = $(PLAT_LINKS:%=$(USR_PLAT_DIR)/%)
USR_PLAT_LINKS_2         = $(PLAT_LINKS_2:%=$(USR_PLAT_DIR)/%)

#
# Collection of all relevant, delivered kernel modules.
#
# Note that we insist on building genunix first, because everything else
# uniquifies against it.  When doing a 'make' from usr/src/uts/, we'll enter
# the platform directories first.  These will cd into the corresponding genunix
# directory and build it.  So genunix /shouldn't/ get rebuilt when we get to
# building all the kernel modules.  However, due to an as-yet-unexplained
# problem with dependencies, sometimes it does get rebuilt, which then messes
# up the other modules.  So we always force the issue here rather than try to
# build genunix in parallel with everything else.
#
PARALLEL_KMODS = $(DRV_KMODS) $(EXEC_KMODS) $(FS_KMODS) $(SCHED_KMODS) \
                 $(TOD_KMODS) $(STRMOD_KMODS) $(SYS_KMODS) $(MISC_KMODS) \
                 $(NLMISC_KMODS) $(MACH_KMODS) $(CPU_KMODS) $(GSS_KMODS) \
                 $(MMU_KMODS) $(DACF_KMODS) $(EXPORT_KMODS) $(IPP_KMODS) \
                 $(CRYPTO_KMODS) $(PCBE_KMODS) \
                 $(DRV_KMODS_$(CLASS)) $(MISC_KMODS_$(CLASS)) $(MAC_KMODS) \
                 $(BRAND_KMODS) $(KICONV_KMODS) \
                 $(SOCKET_KMODS)

KMODS = $(GENUNIX_KMODS) $(PARALLEL_KMODS)

$(PARALLEL_KMODS): $(GENUNIX_KMODS)

$(CLOSED_BUILD)CLOSED_KMODS = $(CLOSED_DRV_KMODS) $(CLOSED_TOD_KMODS) \
        $(CLOSED_MISC_KMODS) $(CLOSED_CPU_KMODS) \
        $(CLOSED_NLMISC_KMODS) $(CLOSED_DRV_KMODS_$(CLASS))

LINT_KMODS = $(DRV_KMODS) $(EXEC_KMODS) $(FS_KMODS) $(SCHED_KMODS) \
          $(TOD_KMODS) $(STRMOD_KMODS) $(SYS_KMODS) $(MISC_KMODS) \
          $(MACH_KMODS) $(GSS_KMODS) $(DACF_KMODS) $(IPP_KMODS) \
          $(CRYPTO_KMODS) $(PCBE_KMODS) \
          $(DRV_KMODS_$(CLASS)) $(MISC_KMODS_$(CLASS)) $(MAC_KMODS) \
          $(BRAND_KMODS) $(KICONV_KMODS) $(SOCKET_KMODS)

$(CLOSED_BUILD)CLOSED_LINT_KMODS = $(CLOSED_DRV_KMODS) $(CLOSED_TOD_KMODS) \
          $(CLOSED_MISC_KMODS) $(CLOSED_DRV_KMODS_$(CLASS))

THIS_YEAR:sh=   /bin/date +%Y
$(OBJS_DIR)/logsubr.o   := CPPFLAGS += -DTHIS_YEAR=$(THIS_YEAR)
$(OBJS_DIR)/logsubr.ln  := CPPFLAGS += -DTHIS_YEAR=$(THIS_YEAR)

#
```

```
641 #       Files to be compiled with -xa, to generate basic block execution
642 #       count data.
643 #
644 #       There are several ways to compile parts of the kernel for kcov:
645 #               1)  Add targets to BB_FILES here or in other Makefiles
646 #                       (they must in the form of $(OBJS_DIR)/target.o)
647 #               2)  setenv BB_FILES '$(XXX_OBJS:%=$(OBJS_DIR)/%)'
648 #               3)  setenv BB_FILES '$(OBJECTS)'
649 #
650 #       Do NOT setenv CFLAGS -xa, as that will cause infinite recursion
651 #       in unix_bb.o
652 #
653 BB_FILES =
654 $(BB_FILES)     := XAOPT = -xa

656 #
657 #       The idea here is for unix_bb.o to be in all kernels except the
658 #       kernel which actually gets shipped to customers.  In practice,
659 #       $(RELEASE_BUILD) is on for a number of the late beta and fcs builds.
660 #
661 $(NOT_RELEASE_BUILD)$(OBJS_DIR)/unix_bb.o   := CPPFLAGS     += -DKCOV
662 $(NOT_RELEASE_BUILD)$(OBJS_DIR)/unix_bb.ln  := CPPFLAGS     += -DKCOV

664 #
665 #       Do not let unix_bb.o get compiled with -xa!
666 #
667 $(OBJS_DIR)/unix_bb.o   := XAOPT =

669 #
670 # Privilege files
671 #
672 PRIVS_AWK = $(SRC)/uts/common/os/privs.awk
673 PRIVS_DEF = $(SRC)/uts/common/os/priv_defs

675 #
676 # USB device data
677 #
678 USBDEVS_AWK =   $(SRC)/uts/common/io/usb/usbdevs2h.awk
679 USBDEVS_DATA =  $(SRC)/uts/common/io/usb/usbdevs
```

```
**********************************************************
    1482 Wed Oct 16 17:41:08 2013
new/usr/src/uts/common/nfs/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# uts/common/nfs/Makefile
#
# include global definitions
#
include ../../../Makefile.master

HDRS=   export.h        lm.h                                            \
        mount.h         nfs.h           nfssys.h        nfs_acl.h       \
        nfs_clnt.h      nfs_log.h       nfs_sec.h       nfs4.h          \
        nfs4_attr.h     nfs4_clnt.h     rnode.h         rnode4.h        \
        nfs4_kprot.h    nfs4_db_impl.h  nfs4_idmap_impl.h \
        nfsid_map.h     auth.h          nfs_cmd.h

$(CLOSED_BUILD)CLOSEDHDRS=      lm_impl.h       lm_nlm.h        lm_server.h

ALLHDRS= $(HDRS) $(CLOSEDHDRS)

ROOTDIRS= $(ROOT)/usr/include/nfs

ROOTHDRS= $(HDRS:%=$(ROOTDIRS)/%)
ROOTHDRS= $(ALLHDRS:%=$(ROOTDIRS)/%)

CHECKHDRS= $(HDRS:%.h=%.check)
CHECKHDRS= $(HDRS:%.h=%.check) \
        $(CLOSEDHDRS:%.h=$(CLOSED)/uts/common/nfs/%.check)

$(ROOTDIRS)/%: %
        $(INS.file)

$(ROOTDIRS)/%: $(CLOSED)/uts/common/nfs/%
        $(INS.file)

.KEEP_STATE:
```



```
.PARALLEL: $(CHECKHDRS)

install_h: $(ROOTDIRS) $(ROOTHDRS)

$(ROOTDIRS):
        $(INS.dir)

check:  $(CHECKHDRS)
```

```
**********************************************************
    4175 Wed Oct 16 17:41:08 2013
new/usr/src/uts/i86pc/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#       This makefile drives the production of all implementation architecture
#       dependent modules for the i86pc architecture.
#

UTSBASE = ..

include Makefile.i86pc

#
#       The following are x86 specific (rather than i86pc) specific modules
#       which are required for the i86pc kernel to completely lint. They are
#       not involved in the build in any other way. In order to minimize
#       build time, it is assumed that they are up to date.
#
INTEL_LIB_DIR    = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)

INTEL_LINTS      = genunix

LINT_LIBS        = \
                   $(GENUNIX_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(PARALLEL_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(CLOSED_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(INTEL_LINTS:%=$(INTEL_LIB_DIR)/llib-l%.ln)

I86PC_LINTS =   dr drmach_acpi

#
#
#
def             :=      TARGET= def
all             :=      TARGET= all
install         :=      TARGET= install
install_h       :=      TARGET= install_h
clean           :=      TARGET= clean
```



```
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint
lintlib         :=      TARGET= lintlib
machmodlintlib  :=      TARGET= modlintlib
modlist         :=      TARGET= modlist
modlist modlist.intel :=        NO_STATE= -K $$MODSTATE$$$$
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check

.KEEP_STATE:

.PARALLEL:      $(PARALLEL_KMODS) $(XMODS) modlist modlist.intel
.PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
                modlist modlist.intel

INITIAL_TARGETS = \
        genassym \
        unix \
        cpu/scripts

def all clean clobber clean.lint: setup genassym unix .WAIT \
        $(KMODS) $(XMODS) $(IMPLEMENTATIONS)
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) $(IMPLEMENTATIONS)

install: install_platforms setup genassym unix .WAIT \
        $(KMODS) $(XMODS) $(IMPLEMENTATIONS)
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) $(IMPLEMENTATIONS)

# Need to clean in here too because of lint.
clean: $(I86PC_LINTS)

# list the modules under i86pc.
modlist: unix $(KMODS) $(XMODS) $(IMPLEMENTATIONS)
modlist: unix $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
        $(IMPLEMENTATIONS)

# list the modules for Install -k i86pc.
modlist.karch: modlist modlist.intel

modlist.intel:
        @cd $(SRC)/uts/intel; pwd; $(MAKE) $(NO_STATE) modlist

lintlib:        unix

modlintlib:     $(KMODS)
modlintlib:     $(KMODS) $(CLOSED_KMODS)

genassym unix $(KMODS): FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

setup: FRC
        @cd cpu/scripts; pwd; $(MAKE) $(TARGET)

$(IMPLEMENTATIONS):     FRC
        @cd $@; pwd; THISIMPL=$@ $(MAKE) $(NO_STATE) $(TARGET)

$(XMODS):       FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

$(CLOSED_KMODS):        FRC
        cd $(CLOSED)/uts/i86pc/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)
```

```
$(CLOSED_XMODS):        FRC
        @if [ -f $(CLOSED)/uts/i86pc/$@/Makefile  ]; then \
                cd $(CLOSED)/uts/i86pc/$@; pwd; \
                    $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

install_h check:        $(IMPLEMENTATIONS) FRC
        @cd sys; pwd; $(MAKE) $(TARGET)

#
# Definitions for the /platform directory aliases.
# Currently none for i86pc.
#
PLAT_LINKS      =

#
# Make the /platform directories.  This is hardwired here because
# the first stage of the project (KBI) only implements the userland
# changes, but the only reasonable place to record the aliases is
# here in kernel land.
#
install_platforms:      $(ROOT_PSM_DIR) $(USR_PSM_DIR) \
                        $(ROOT_PLAT_LINKS) $(USR_PLAT_LINKS) \
                        $(OEM_USR_PLAT_LINKS)

#
#       Full kernel lint target.
#
LINT_TARGET     = globallint

# workaround for multiply defined errors
globallint := LINTFLAGS += -erroff=E_NAME_MULTIPLY_DEF2

globallint:
        @-$(ECHO) "\nFULL KERNEL: global crosschecks:"
        @-$(LINT) $(LINTFLAGS) $(LINT_LIB) $(LINT_LIBS) 2>&1 | $(LGREP.2)

lint:   lintlib .WAIT modlintlib .WAIT $(INTEL_LINTS) $(LINT_DEPS) \
        $(IMPLEMENTATIONS)

$(INTEL_LINTS): FRC
        @cd $(UTSBASE)/intel/$@; pwd; $(MAKE) modlintlib

include ../Makefile.targ

#
# Cross-reference customization: build a cross-reference over all of the
# i86pc-related directories.
#
XRDIRS  = ../i86pc ../intel ../common
$(CLOSED_BUILD)XRDIRS   += ../../../closed/uts/intel ../../../closed/uts/common

XRPRUNE = sun4u sun4

cscope.out tags: FRC
        $(XREF) -x $@
```

```
**********************************************************
    9085 Wed Oct 16 17:41:08 2013
new/usr/src/uts/i86pc/Makefile.i86pc.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# uts/i86pc/Makefile.i86pc
#
# Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2013 Andrew Stormont.  All rights reserved.
#
#
#       This makefile contains the common definitions for the i86pc unix
#       and all i86pc implementation architecture dependent modules.
#

#
#       Machine type (implementation architecture):
#
PLATFORM         = i86pc

#
#       uname -m value
#
UNAME_M          = $(PLATFORM)

#
# Definitions for the platform-specific /platform directories.
#
# IMPLEMENTATIONS is used to designate i86pc machines which have
# platform specific modules.  All code specific to a given implementation
# resides in the appropriately named subdirectory.   This requires
# these platforms to have their own Makefiles to define ROOT_PLAT_DIRS,
# USR_PLAT_DIRS, etc.
#
IMPLEMENTATIONS = i86hvm

#
#       Everybody needs to know how to build modstubs.o and to locate unix.o
#
UNIX_DIR         = $(UTSBASE)/$(PLATFORM)/unix
GENLIB_DIR       = $(UTSBASE)/intel/genunix
```



```
MODSTUBS_DIR     = $(UNIX_DIR)
DSF_DIR          = $(UTSBASE)/$(PLATFORM)/genassym
LINTS_DIR        = $(OBJS_DIR)
LINT_LIB_DIR     = $(UTSBASE)/$(PLATFORM)/lint-libs/$(OBJS_DIR)
GEN_LINT_LIB_DIR = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)

LINT32_DIRS      = $(LINT32_BUILDS:%=$(UTSBASE)/$(PLATFORM)/lint-libs/%)
LINT32_FILES     = $(LINT32_DIRS:%=%/llib-l$(MODULE).ln)

DTRACESTUBS_O    = $(OBJS_DIR)/dtracestubs.o
DTRACESTUBS      = $(OBJS_DIR)/libdtracestubs.so

SYM_MOD         = $(OBJS_DIR)/unix.sym

UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
GENLIB           = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so
LINT_LIB         = $(LINT_LIB_DIR)/llib-lunix.ln
DBOOT_LINT_LIB   = $(LINT_LIB_DIR)/llib-ldboot.ln
GEN_LINT_LIB     = $(GEN_LINT_LIB_DIR)/llib-lgenunix.ln

#
#       Include the makefiles which define build rule templates, the
#       collection of files per module, and a few specific flags. Note
#       that order is significant, just as with an include path. The
#       first build rule template which matches the files name will be
#       used. By including these in order from most machine dependent
#       to most machine independent, we allow a machine dependent file
#       to be used in preference over a machine independent version
#       (Such as a machine specific optimization, which preserves the
#       interfaces.)
#
include $(UTSTREE)/$(PLATFORM)/Makefile.files
include $(UTSTREE)/intel/Makefile.files
include $(UTSTREE)/common/Makefile.files

#
#       Include machine independent rules. Note that this does not imply
#       that the resulting module from rules in Makefile.uts is machine
#       independent. Only that the build rules are machine independent.
#
include $(UTSBASE)/Makefile.uts

#
#       Define supported builds
#
DEF_BUILDS              = $(DEF_BUILDS64) $(DEF_BUILDS32)
ALL_BUILDS              = $(ALL_BUILDS64) $(ALL_BUILDS32)

#
#       x86 or amd64 inline templates
#
INLINES_32              = $(UTSBASE)/intel/ia32/ml/ia32.il \
                          $(UTSBASE)/$(PLATFORM)/ml/ia32.il
INLINES_64              = $(UTSBASE)/intel/amd64/ml/amd64.il \
                          $(UTSBASE)/$(PLATFORM)/ml/amd64.il
INLINES                 += $(INLINES_$(CLASS))

#
#       kernel-specific optimizations; override default in Makefile.master
#

CFLAGS_XARCH_32         = $(i386_CFLAGS)
CFLAGS_XARCH_64         = $(amd64_CFLAGS)
CFLAGS_XARCH            = $(CFLAGS_XARCH_$(CLASS))
```

```
125 COPTFLAG_32             = $(COPTFLAG)
126 COPTFLAG_64             = $(COPTFLAG64)
127 COPTIMIZE               = $(COPTFLAG_$(CLASS))

129 CFLAGS                  = $(CFLAGS_XARCH)
130 CFLAGS                  += $(COPTIMIZE)
131 CFLAGS                  += $(INLINES) -D_ASM_INLINES
132 CFLAGS                  += $(CCMODE)
133 CFLAGS                  += $(SPACEFLAG)
134 CFLAGS                  += $(CCUNBOUND)
135 CFLAGS                  += $(CFLAGS_uts)
136 CFLAGS                  += -xstrconst

138 ASFLAGS_XARCH_32        = $(i386_ASFLAGS)
139 ASFLAGS_XARCH_64        = $(amd64_ASFLAGS)
140 ASFLAGS_XARCH           = $(ASFLAGS_XARCH_$(CLASS))

142 ASFLAGS                 += $(ASFLAGS_XARCH)

144 AS_INC_PATH             += -I$(DSF_DIR)/$(OBJS_DIR)

146 #
147 #       The following must be defined for all implementations:
148 #
149 #       MAPFILE:        ld mapfile for the build of kernel/unix.
150 #       MODSTUBS:       Module stubs source file.
151 #       GENASSYM_SRC:   genassym.c
152 #
153 MAPFILE          = $(UTSBASE)/$(PLATFORM)/conf/Mapfile
154 MODSTUBS         = $(UTSBASE)/intel/ia32/ml/modstubs.s
155 GENASSYM_SRC     = $(UTSBASE)/$(PLATFORM)/ml/genassym.c
156 OFFSETS_SRC      = $(UTSBASE)/$(PLATFORM)/ml/offsets.in
157 PLATFORM_OFFSETS_32     = $(UTSBASE)/$(PLATFORM)/ml/mach_offsets.in
158 PLATFORM_OFFSETS_64     = $(UTSBASE)/intel/amd64/ml/mach_offsets.in
159 PLATFORM_OFFSETS_SRC    = $(PLATFORM_OFFSETS_$(CLASS))
160 KDI_OFFSETS_SRC  = $(UTSBASE)/intel/kdi/kdi_offsets.in

162 #
163 #       Define the actual specific platforms
164 #
165 MACHINE_DEFS     = -D$(PLATFORM) -D_MACHDEP

167 #
168 #       Software workarounds for hardware "features"
169 #

171 include $(UTSBASE)/$(PLATFORM)/Makefile.workarounds

173 #
174 #       Debugging level
175 #
176 #       Special knowledge of which special debugging options effect which
177 #       file is used to optimize the build if these flags are changed.
178 #
179 #       XXX: The above could possibly be done for more flags and files, but
180 #            is left as an experiment to the interested reader. Be forewarned,
181 #            that excessive use could lead to maintenance difficulties.
182 #
183 DEBUG_DEFS_OBJ32        =
184 DEBUG_DEFS_DBG32        = -DDEBUG
185 DEBUG_DEFS_OBJ64        =
186 DEBUG_DEFS_DBG64        = -DDEBUG
187 DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

189 DEBUG_COND_OBJ32        = $(POUND_SIGN)
190 DEBUG_COND_DBG32        =
```



```
191 DEBUG_COND_OBJ64        = $(POUND_SIGN)
192 DEBUG_COND_DBG64        =
193 IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

195 $(IF_DEBUG_OBJ)trap.o           := DEBUG_DEFS += -DTRAPDEBUG -DTRAPTRACE
196 $(IF_DEBUG_OBJ)syscall_asm.o    := DEBUG_DEFS += -DSYSCALLTRACE -DTRAPTRACE
197 $(IF_DEBUG_OBJ)syscall_asm_amd64.o := DEBUG_DEFS += -DSYSCALLTRACE -DTRAPTRACE
198 $(IF_DEBUG_OBJ)fast_trap_asm.o  := DEBUG_DEFS += -DTRAPTRACE
199 $(IF_DEBUG_OBJ)interrupt.o      := DEBUG_DEFS += -DTRAPTRACE
200 $(IF_DEBUG_OBJ)intr.o           := DEBUG_DEFS += -DTRAPTRACE
201 $(IF_DEBUG_OBJ)locore.o         := DEBUG_DEFS += -DTRAPTRACE
202 $(IF_DEBUG_OBJ)mp_startup.o     := DEBUG_DEFS += -DTRAPTRACE
203 $(IF_DEBUG_OBJ)machdep.o        := DEBUG_DEFS += -DTRAPTRACE
204 $(IF_DEBUG_OBJ)exception.o      := DEBUG_DEFS += -DTRAPTRACE
205 $(IF_DEBUG_OBJ)x_call.o         := DEBUG_DEFS += -DTRAPTRACE
206 $(IF_DEBUG_OBJ)mp_call.o        := DEBUG_DEFS += -DTRAPTRACE
207 $(IF_DEBUG_OBJ)cbe.o            := DEBUG_DEFS += -DTRAPTRACE

209 #
210 #       Collect the preprocessor definitions to be associated with *all*
211 #       files.
212 #
213 ALL_DEFS         = $(MACHINE_DEFS) $(WORKAROUND_DEFS) $(DEBUG_DEFS) \
214                    $(OPTION_DEFS)
215 GENASSYM_DEFS    = $(MACHINE_DEFS) $(OPTION_DEFS) \
216                         -_gcc=-fno-eliminate-unused-debug-symbols \
217                         -_gcc=-fno-eliminate-unused-debug-types

219 #
220 # ----- TRANSITIONAL SECTION --------------------------------------------------
221 #

223 #
224 #       Not everything which *should* be a module is a module yet. The
225 #       following is a list of such objects which are currently part of
226 #       the base kernel but should soon become kmods.
227 #
228 #       XXX: $(KMACCT_OBJS) is neither in the MT kernel nor was it ever
229 #            made into a module. If it is made MT safe before being made
230 #            into a module, it should be added to this list. It was in
231 #            this list pre ON-4.0.
232 #
233 #
234 MACH_NOT_YET_KMODS      = $(AUTOCONF_OBJS)

236 #
237 # ----- END OF TRANSITIONAL SECTION -------------------------------------------
238 #

240 #
241 #       The kernels modules which are "implementation architecture"
242 #       specific for this machine are enumerated below. Note that most
243 #       of these modules must exist (in one form or another) for each
244 #       architecture.
245 #
246 #       Machine Specific Driver Modules (/kernel/drv)
247 #       DRV_KMODS are built both 32-bit and 64-bit
248 #       DRV_KMODS_32 are built only 32-bit
249 #       DRV_KMODS_64 are built only 64-bit
250 #
251 DRV_KMODS       += rootnex
252 DRV_KMODS       += isa
253 DRV_KMODS       += pcplusmp
254 DRV_KMODS       += apix
255 DRV_KMODS       += cpc
256 DRV_KMODS       += pci
```

```
257 DRV_KMODS       += npe
258 DRV_KMODS       += pci-ide
259 DRV_KMODS       += xsvc
260 DRV_KMODS       += tzmon
261 DRV_KMODS       += acpi_drv
262 DRV_KMODS       += acpinex
263 DRV_KMODS       += amd_iommu
264 DRV_KMODS       += dr
265 DRV_KMODS       += ioat
266 DRV_KMODS       += fipe

268 DRV_KMODS       += cpudrv


271 #
272 # Platform Power Modules
273 #
274 DRV_KMODS       += ppm acpippm

276 #
277 #       CPU Modules
278 #
279 CPU_KMODS       += amd_opteron
280 CPU_KMODS       += generic_cpu
281 CPU_KMODS       += authenticamd
282 CPU_KMODS       += genuineintel

284 #
285 # Don't build some of these for OpenSolaris, since they will be
286 # replaced by binaries that are signed by Sun Release Engineering.
287 #
288 $(CLOSED_BUILD)CLOSED_CPU_KMODS += intel_nhmex

290 #
285 #       Exec Class Modules (/kernel/exec):
286 #
287 EXEC_KMODS      +=

289 #
290 #       Scheduling Class Modules (/kernel/sched):
291 #
292 SCHED_KMODS     +=

294 #
295 #       File System Modules (/kernel/fs):
296 #
297 FS_KMODS        +=

299 #
300 #       Streams Modules (/kernel/strmod):
301 #
302 STRMOD_KMODS    +=

304 #
305 #       'System' Modules (/kernel/sys):
306 #
307 SYS_KMODS       +=

309 #
310 #       'Misc' Modules (/kernel/misc):
311 #
312 MISC_KMODS      += gfx_private pcie
313 MISC_KMODS      += acpidev
314 MISC_KMODS      += drmach_acpi

316 #
```



```
317 #       'Dacf' modules (/kernel/dacf)
318 #
319 DACF_KMODS      += consconfig_dacf

321 #
322 #       'Mach' Modules (/kernel/mach):
323 #
324 MACH_KMODS      += uppc

326 #
327 #       CPR Misc Module.
328 #
329 MISC_KMODS      += cpr
```

```
**********************************************************
    4079 Wed Oct 16 17:41:09 2013
new/usr/src/uts/i86xpv/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
#
#       This makefile drives the production of all implementation architecture
#       dependent modules for the i86xpv architecture.
#

UTSBASE = ..

include Makefile.i86xpv

#
#       The following are x86 specific (rather than i86pc) specific modules
#       which are required for the i86pc kernel to completely lint. They are
#       not involved in the build in any other way. In order to minimize
#       build time, it is assumed that they are up to date.
#
INTEL_LIB_DIR    = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)

INTEL_LINTS      = genunix

LINT_LIBS        = \
                   $(GENUNIX_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(PARALLEL_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(CLOSED_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(INTEL_LINTS:%=$(INTEL_LIB_DIR)/llib-l%.ln)

I86XPV_LINTS =  xdb xnb xnbe xnbo xnbu xpvtap

#
#
#
def             :=      TARGET= def
all             :=      TARGET= all
install         :=      TARGET= install
```



```
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint
lintlib         :=      TARGET= lintlib
machmodlintlib  :=      TARGET= modlintlib
modlist         :=      TARGET= modlist
modlist modlist.intel :=        NO_STATE= -K $$MODSTATE$$$$
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check

.KEEP_STATE:

.PARALLEL:      $(PARALLEL_KMODS) $(XMODS) modlist modlist.intel
.PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
                modlist modlist.intel

INITIAL_TARGETS = \
        genassym \
        unix \
        cpu/scripts

def all clean clobber clean.lint: setup genassym unix .WAIT \
        $(KMODS) $(XMODS)
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS)

install: install_platforms setup genassym unix .WAIT \
        $(KMODS) $(XMODS)
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS)

# Need to clean in here too because of lint.
clean: $(I86XPV_LINTS)

# list the modules under i86xpv.
modlist: unix $(KMODS) $(XMODS)
modlist: unix $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS)

# list the modules for Install -k i86xpv.
modlist.karch: modlist modlist.intel

modlist.intel:
        @cd $(SRC)/uts/intel; pwd; $(MAKE) $(NO_STATE) modlist

lintlib:        unix

modlintlib:     $(KMODS)
modlintlib:     $(KMODS) $(CLOSED_KMODS)

genassym unix $(KMODS): FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

setup: FRC
        @cd cpu/scripts; pwd; $(MAKE) $(TARGET)

$(XMODS):       FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

$(CLOSED_KMODS):        FRC
        cd $(CLOSED)/uts/i86xpv/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

$(CLOSED_XMODS):        FRC
        @if [ -f $(CLOSED)/uts/i86xpv/$@/Makefile  ]; then \
                cd $(CLOSED)/uts/i86xpv/$@; pwd; \
```

```
120                     $(MAKE) $(NO_STATE) $(TARGET); \
121         else \
122                 true; \
123         fi

112 install_h check:        FRC
113         @cd sys; pwd; $(MAKE) $(TARGET)

115 #
116 # Definitions for the /platform directory aliases.
117 # Currently none for i86xpv.
118 #
119 PLAT_LINKS      =

121 #
122 # Make the /platform directories.  This is hardwired here because
123 # the first stage of the project (KBI) only implements the userland
124 # changes, but the only reasonable place to record the aliases is
125 # here in kernel land.
126 #
127 install_platforms:      $(ROOT_PSM_DIR) $(USR_PSM_DIR) \
128                         $(ROOT_PLAT_LINKS) $(USR_PLAT_LINKS) \
129                         $(OEM_USR_PLAT_LINKS)

131 #
132 #       Full kernel lint target.
133 #
134 LINT_TARGET     = globallint

136 # workaround for multiply defined errors
137 globallint := LINTFLAGS += -erroff=E_NAME_MULTIPLY_DEF2

139 globallint:
140         @-$(ECHO) "\nFULL KERNEL: global crosschecks:"
141         @-$(LINT) $(LINTFLAGS) $(LINT_LIB) $(LINT_LIBS) 2>&1 | $(LGREP.2)

143 lint:   lintlib .WAIT modlintlib .WAIT $(INTEL_LINTS) $(LINT_DEPS)

145 $(INTEL_LINTS): FRC
146         @cd $(UTSBASE)/intel/$@; pwd; $(MAKE) modlintlib

148 FRC:

150 include ../Makefile.targ

152 #
153 # Cross-reference customization: build a cross-reference over all of the
154 # i86pc-related directories.
155 #
156 XRDIRS  = ../i86xpv ../i86pc ../intel ../common
169 SHARED_XRDIRS   = ../i86xpv ../i86pc ../intel ../common
170 XRDIRS          = $(SHARED_XRDIRS)
171 CLOSED_XRDIRS1  = $(SHARED_XRDIRS:../%=../% ../../../closed/uts/%)
172 CLOSED_XRDIRS2  = $(CLOSED_XRDIRS1:../../../closed/uts/i86pc=)
173 $(CLOSED_BUILD)XRDIRS   = $(CLOSED_XRDIRS2:../../../closed/uts/i86xpv=)
157 XRPRUNE = sun4v sun4u sun4 sfmmu sparc

159 cscope.out tags: FRC
160         $(XREF) -x $@
```

```
**********************************************************
    4503 Wed Oct 16 17:41:09 2013
new/usr/src/uts/intel/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 # uts/intel/Makefile
  22 #
  23 # Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
  24 #
  25 #       This makefile drives the production of all implementation architecture
  26 #       independent modules for Intel processors.

  28 UTSBASE = ..

  30 include Makefile.intel

  32 LINT_KMODS_X1   = $(LINT_KMODS:nsmb=)
  33 LINT_KMODS_X2   = $(LINT_KMODS_X1:smbfs=)
  34 LINT_KMODLIBS   = $(LINT_KMODS_X2:e1000g=)
  35 LINT_LIBS       = $(LINT_LIB) $(GEN_LINT_LIB) \
  36                   $(LINT_KMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln)
  36                   $(LINT_KMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
  37                   $(CLOSED_LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln)

  39 $(CLOSED_BUILD)LINT_LIBS        += $(SVVS_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln)
  40 $(CLOSED_BUILD)LINT_CLOSED_XMOD4         = $(CLOSED_XMODS:bnx=)
  41 $(CLOSED_BUILD)LINT_CLOSED_XMOD3         = $(LINT_CLOSED_XMOD4:bnxe=)
  42 $(CLOSED_BUILD)LINT_CLOSED_XMOD2         = $(LINT_CLOSED_XMOD3:lsimega=)
  43 $(CLOSED_BUILD)LINT_CLOSED_XMOD1         = $(LINT_CLOSED_XMOD2:adpu320=)
  44 $(CLOSED_BUILD)LINT_LIBS        += $(LINT_XMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln)

  38 #
  39 # dprov is delivered in the SUNWcryptoint package.
  40 #
  41 DRV_KMODS       += dprov

  43 #
  44 #
  45 def             :=      TARGET= def
  46 def.prereq      :=      TARGET= def
  47 all             :=      TARGET= all
  48 all.prereq      :=      TARGET= all
  49 install         :=      TARGET= install
```



```
  50 install.prereq  :=      TARGET= all
  51 clean           :=      TARGET= clean
  52 clobber         :=      TARGET= clobber
  53 lint            :=      TARGET= lint
  54 lint.prereq     :=      TARGET= lint
  55 modlintlib      :=      TARGET= modlintlib
  56 modlist         :=      TARGET= modlist
  57 modlist         :=      NO_STATE= -K $$MODSTATE$$$$
  58 clean.lint      :=      TARGET= clean.lint
  59 check           :=      TARGET= check
  60 install_h       :=      TARGET= install_h
  61 install_h.prereq        :=      TARGET= install_h

  63 .KEEP_STATE:

  65 .PARALLEL:      $(PARALLEL_KMODS) $(XMODS) config $(LINT_DEPS)
  73 .PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(SVVS) $(XMODS) \
  74                 $(CLOSED_XMODS) config $(LINT_DEPS)

  67 def all install clean clobber modlist: $(KMODS) $(XMODS) config
  76 def all install clean clobber modlist: $(KMODS) $(CLOSED_KMODS) \
  77         $(SVVS) $(XMODS) $(CLOSED_XMODS) config

  69 clobber: clobber.targ

  71 #
  72 # Privilege constants
  73 #
  74 # NOTE: The rules for generating priv_const.c file are shared between all
  75 # processor architectures and and should be kept in sync. If they are changed in
  76 # this file make sure that sparc rules are updated as well.
  77 #
  78 PRIVS_C = $(SRC)/uts/common/os/priv_const.c

  80 $(PRIVS_C): $(PRIVS_AWK) $(PRIVS_DEF)
  81         $(NAWK) -f $(PRIVS_AWK) < $(PRIVS_DEF) cfile=$@

  83 CLOBBERFILES += $(PRIVS_C)

  85 #
  86 # Prerequisites
  87 #
  88 # The uts/Makefile defines build parallelism for x86 platforms such that i86pc,
  89 # i86xpv and intel are all built in parallel. This requires building certain
  90 # parts before the parallel build can start. The uts/Makefile appends the
  91 # '.prereq' string to the original target and executes this Makefile to build
  92 # any prerequisites needed before the full parallel build can start. After that
  93 # make continues with normal targets.
  94 #
  95 # Any build prerequisites for x86 builds should be described here.
  96 #
  97 # genassym is used to build intel/dtrace and genunix, so it should be built
  98 # first.
  99 #
 100 # priv_const.c is required to build genunix.
 101 #
 102 # genunix is used by everyone to ctf-merge with. Genunix is CTF-merged with
 103 #   intel/ip so as a side effect this dependency builds intel/ip as part of the
 104 #   prerequisites.
 105 #
 106 # intel/dtrace depends on i86pc/genassym, so we need to build both
 107 # i86pc/genassym and intel/genassym.
 108 #
 109 all.prereq install.prereq def.prereq: genunix FRC
 110         @cd ../i86pc/genassym; pwd; $(MAKE) $(@:%.prereq=%)
```

```
112 #
113 # i86pc lint libraries should be built first
114 #
115 lint.prereq: FRC
116         @cd ../i86pc; pwd; $(MAKE) $(NO_STATE) lint

118 #
119 # Nothing to do for any other prerequisite targets.
120 #
121 %.prereq:

123 genunix: $(PRIVS_C)

125 modlintlib clean.lint: $(LINT_KMODS) $(XMODS)
135 modlintlib clean.lint: $(LINT_KMODS) $(CLOSED_LINT_KMODS) $(SVVS) \
136         $(XMODS) $(CLOSED_XMODS)

127 $(KMODS) $(SUBDIRS) config:     FRC
128         @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

141 $(CLOSED_KMODS):        FRC
142         cd $(CLOSED)/uts/intel/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

130 $(XMODS):       FRC
131         @if [ -f $@/Makefile  ]; then \
132                 cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
133         else \
134                 true; \
135         fi

151 $(SVVS) $(CLOSED_XMODS):        FRC
152         @if [ -f $(CLOSED)/uts/intel/$@/Makefile  ]; then \
153                 cd $(CLOSED)/uts/intel/$@; pwd; \
154                     $(MAKE) $(NO_STATE) $(TARGET); \
155         else \
156                 true; \
157         fi

137 install_h check:        FRC
138         @cd sys; pwd; $(MAKE) $(TARGET)
139         @cd asm; pwd; $(MAKE) $(TARGET)
140         @cd ia32/sys; pwd; $(MAKE) $(TARGET)
141         @cd amd64/sys; pwd; $(MAKE) $(TARGET)

143 #
144 # Work-around to disable acpica global crosscheck lint warnings
145 #
146 LGREP.intel =   grep -v 'intel/io/acpica'

148 #
149 #       Full kernel lint target.
150 #
151 LINT_TARGET     = globallint

153 # workaround for multiply defined errors
154 globallint := LINTFLAGS += -erroff=E_NAME_MULTIPLY_DEF2

156 globallint:
157         @pwd
158         @-$(ECHO) "\nFULL KERNEL: global crosschecks:"
159         @-$(LINT) $(LINTFLAGS) $(LINT_LIBS) 2>&1 | $(LGREP.intel) | $(LGREP.2)

161 lint:   modlintlib .WAIT $(LINT_DEPS)

163 include ../Makefile.targ
```

```
**********************************************************
    6816 Wed Oct 16 17:41:09 2013
new/usr/src/uts/intel/Makefile.files
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2012, Joyent, Inc. All rights reserved.
#

#
#       This Makefile defines all file modules and build rules for the
# directory uts/intel and its children. These are the source files which
# are specific to x86 processor architectures.
#

#
#       Core (unix) objects
#
CORE_OBJS +=            \
        arch_kdi.o      \
        copy.o          \
        copy_subr.o     \
        cpc_subr.o      \
        ddi_arch.o      \
        ddi_i86.o       \
        ddi_i86_asm.o   \
        desctbls.o      \
        desctbls_asm.o  \
        exception.o     \
        float.o         \
        fmsmb.o         \
        fpu.o           \
        i86_subr.o      \
        lock_prim.o     \
        ovbcopy.o       \
        polled_io.o     \
        sseblk.o        \
        sundep.o        \
        swtch.o         \
        sysi86.o
```



```
#
# 64-bit multiply/divide compiler helper routines
# used only for ia32
#

SPECIAL_OBJS_32 +=      \
        muldiv.o

#
#       Generic-unix Module
#
GENUNIX_OBJS +=         \
        archdep.o       \
        getcontext.o    \
        install_utrap.o \
        lwp_private.o   \
        prom_enter.o    \
        prom_exit.o     \
        prom_panic.o    \
        sendsig.o       \
        syscall.o


#
#       PROM Routines
#
GENUNIX_OBJS +=         \
        prom_env.o      \
        prom_emul.o     \
        prom_getchar.o  \
        prom_init.o     \
        prom_node.o     \
        prom_printf.o   \
        prom_prop.o     \
        prom_putchar.o  \
        prom_reboot.o   \
        prom_version.o

#
#       file system modules
#
CORE_OBJS +=            \
        prmachdep.o

#
#       ZFS file system module
#
ZFS_OBJS +=             \
        spa_boot.o

#
#       Decompression code
#
CORE_OBJS += decompress.o

#
#       Microcode utilities
#
CORE_OBJS += ucode_utils.o

#
#       Driver modules
#
AGPGART_OBJS += agpgart.o agp_kstat.o
AGPTARGET_OBJS += agptarget.o
AMD64GART_OBJS += amd64_gart.o
```

```
125 ARCMSR_OBJS += arcmsr.o
126 ATA_OBJS += $(GHD_OBJS) ata_blacklist.o ata_common.o ata_disk.o \
127         ata_dma.o atapi.o atapi_fsm.o ata_debug.o \
128         sil3xxx.o
129 BSCBUS_OBJS += bscbus.o
130 BSCV_OBJS += bscv.o
131 CMDK_OBJS += cmdk.o
132 CMLB_OBJS += cmlb.o
133 CPUNEX_OBJS += cpunex.o
134 DADK_OBJS += dadk.o
135 DCOPY_OBJS += dcopy.o
136 DNET_OBJS += dnet.o dnet_mii.o
137 FD_OBJS += fd.o
138 GDA_OBJS += gda.o
139 GHD_OBJS += ghd.o ghd_debug.o ghd_dma.o ghd_queue.o ghd_scsa.o \
140         ghd_scsi.o ghd_timer.o ghd_waitq.o ghd_gcmd.o
141 I915_OBJS += i915_dma.o i915_drv.o i915_irq.o i915_mem.o \
142         i915_gem.o i915_gem_debug.o i915_gem_tiling.o
143 NSKERN_OBJS += nsc_asm.o
144 PCICFG_OBJS += pcicfg.o
145 PCI_PCINEXUS_OBJS += pci_pci.o
146 PCIEB_OBJS += pcieb_x86.o
147 PIT_BEEP_OBJS += pit_beep.o
148 POWER_OBJS += power.o
149 PCI_AUTOCONFIG_OBJS += pci_autoconfig.o pci_boot.o pcie_nvidia.o \
150                        pci_memlist.o pci_resource.o
151 RADEON_OBJS += r300_cmdbuf.o radeon_cp.o radeon_drv.o \
152                        radeon_state.o radeon_irq.o radeon_mem.o
153 SD_OBJS += sd.o sd_xbuf.o

155 HECI_OBJS +=            \
156         heci_init.o     \
157         heci_intr.o     \
158         heci_interface.o \
159         io_heci.o       \
160         heci_main.o

162 STRATEGY_OBJS += strategy.o
163 UCODE_OBJS += ucode_drv.o
164 VGATEXT_OBJS += vgatext.o vgasubr.o

166 #
167 #       Kernel linker
168 #
169 KRTLD_OBJS +=           \
170         bootrd.o        \
171         ufsops.o        \
172         hsfs.o          \
173         doreloc.o       \
174         kobj_boot.o     \
175         kobj_convrelstr.o \
176         kobj_crt.o      \
177         kobj_isa.o      \
178         kobj_reloc.o

180 #
181 #       misc. modules
182 #
183 ACPICA_OBJS     += dbcmds.o dbdisply.o \
184                    dbexec.o dbfileio.o dbhistry.o dbinput.o dbstats.o \
185                    dbutils.o dbxface.o evevent.o evgpe.o evgpeblk.o \
186                    evmisc.o evregion.o evrgnini.o evsci.o evxface.o \
187                    evxfevnt.o evxfregn.o hwacpi.o hwgpe.o hwregs.o \
188                    hwsleep.o hwtimer.o dsfield.o dsinit.o dsmethod.o \
189                    dsmthdat.o dsobject.o dsopcode.o dsutils.o dswexec.o \
190                    dswload.o dswscope.o dswstate.o exconfig.o exconvrt.o \
```



```
191                    excreate.o exdump.o exfield.o exfldio.o exmisc.o \
192                    exmutex.o exnames.o exoparg1.o exoparg2.o exoparg3.o \
193                    exoparg6.o exprep.o exregion.o exresnte.o exresolv.o \
194                    exresop.o exstore.o exstoren.o exstorob.o exsystem.o \
195                    exutils.o psargs.o psopcode.o psparse.o psscope.o \
196                    pstree.o psutils.o pswalk.o psxface.o nsaccess.o \
197                    nsalloc.o nsdump.o nsdumpdv.o nseval.o nsinit.o \
198                    nsload.o nsnames.o nsobject.o nsparse.o nssearch.o \
199                    nsutils.o nswalk.o nsxfeval.o nsxfname.o nsxfobj.o \
200                    rsaddr.o rscalc.o rscreate.o rsdump.o \
201                    rsinfo.o rsio.o rsirq.o rslist.o rsmemory.o rsmisc.o \
202                    rsutils.o rsxface.o tbfadt.o tbfind.o tbinstal.o  \
203                    tbutils.o tbxface.o tbxfroot.o \
204                    utalloc.o utclib.o utcopy.o utdebug.o utdelete.o \
205                    uteval.o utglobal.o utinit.o utmath.o utmisc.o \
206                    utobject.o utresrc.o utxface.o acpica.o acpi_enum.o \
207                    master_ops.o osl.o osl_ml.o acpica_ec.o utcache.o \
208                    utmutex.o utstate.o dmbuffer.o dmnames.o dmobject.o \
209                    dmopcode.o dmresrc.o dmresrcl.o dmresrcs.o dmutils.o \
210                    dmwalk.o psloop.o nspredef.o hwxface.o hwvalid.o \
211                    utlock.o utids.o nsrepair.o nsrepair2.o \
212                    dbmethod.o dbnames.o dsargs.o dscontrol.o dswload2.o \
213                    evglock.o evgpeinit.o evgpeutil.o evxfgpe.o exdebug.o \
214                    hwpci.o utdecode.o utosi.o utxferror.o


217 AGP_OBJS += agpmaster.o
218 FBT_OBJS += fbt.o
219 SDT_OBJS += sdt.o

221 #
222 #       AMD8111 NIC driver module
223 #
224 AMD8111S_OBJS += amd8111s_main.o amd8111s_hw.o

226 #
227 #       Pentium Performance Counter BackEnd module
228 #
229 P123_PCBE_OBJS = p123_pcbe.o

231 #
232 #       Pentium 4 Performance Counter BackEnd module
233 #
234 P4_PCBE_OBJS = p4_pcbe.o

236 #
237 #       AMD Opteron/Athlon64 Performance Counter BackEnd module
238 #
239 OPTERON_PCBE_OBJS = opteron_pcbe.o

241 #
242 #       Intel Core Architecture Performance Counter BackEnd module
243 #
244 CORE_PCBE_OBJS = core_pcbe.o

246 #
247 #       AMR module
248 #
249 AMR_OBJS = amr.o

251 #
252 #       IPMI module
253 IPMI_OBJS +=    ipmi_main.o ipmi.o ipmi_kcs.o

255 #
256 #       IOMMULIB module
```

```
257 #
258 IOMMULIB_OBJS = iommulib.o

260 #
261 #       Brand modules
262 #
263 SN1_BRAND_OBJS  =       sn1_brand.o sn1_brand_asm.o
264 S10_BRAND_OBJS  =       s10_brand.o s10_brand_asm.o

266 #
267 #       special files
268 #
269 MODSTUB_OBJ +=          \
270         modstubs.o

272 BOOTDEV_OBJS +=         \
273         bootdev.o

275 INC_PATH        += -I$(UTSBASE)/intel


278 CPR_INTEL_OBJS +=       cpr_intel.o

280 #
281 # AMD family 0xf memory controller module
282 #
283 include $(SRC)/common/mc/mc-amd/Makefile.mcamd
284 MCAMD_OBJS      += \
285         $(MCAMD_CMN_OBJS) \
286         mcamd_drv.o \
287         mcamd_dimmcfg.o \
288         mcamd_subr.o \
289         mcamd_pcicfg.o

291 #
292 # Intel Nehalem memory controller module
293 #
294 INTEL_NHM_OBJS += \
295         nhm_init.o \
296         mem_addr.o \
297         intel_nhmdrv.o \
298         nhm_pci_cfg.o \
299         dimm_topo.o \
300         intel_nhm.o

302 #
303 # Intel 5000/5100/5400/7300 chipset memory controller hub (MCH) module
304 #
305 INTEL_NB5000_OBJS += \
306         intel_nb5000.o \
307         intel_nbdrv.o \
308         dimm_addr.o \
309         nb_pci_cfg.o \
310         nb5000_init.o
311 #endif /* ! codereview */
```

```
**********************************************************
   15792 Wed Oct 16 17:41:09 2013
new/usr/src/uts/intel/Makefile.intel.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 # Copyright (c) 2013 Andrew Stormont.  All rights reserved.

  25 #
  26 #       This makefile contains the common definitions for all intel
  27 #       implementation architecture independent modules.
  28 #

  30 #
  31 #       Machine type (implementation architecture):
  32 #
  33 PLATFORM         = i86pc

  35 #
  36 #       Everybody needs to know how to build modstubs.o and to locate unix.o.
  37 #       Note that unix.o must currently be selected from among the possible
  38 #       "implementation architectures". Note further, that unix.o is only
  39 #       used as an optional error check for undefines so (theoretically)
  40 #       any "implementation architectures" could be used. We choose i86pc
  41 #       because it is the reference port.
  42 #
  43 UNIX_DIR         = $(UTSBASE)/i86pc/unix
  44 GENLIB_DIR       = $(UTSBASE)/intel/genunix
  45 IPDRV_DIR        = $(UTSBASE)/intel/ip
  46 MODSTUBS_DIR     = $(UNIX_DIR)
  47 DSF_DIR          = $(UTSBASE)/$(PLATFORM)/genassym
  48 LINTS_DIR        = $(OBJS_DIR)
  49 LINT_LIB_DIR     = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)

  51 UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
  52 GENLIB           = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so
  53 MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
  54 LINT_LIB         = $(UTSBASE)/i86pc/lint-libs/$(OBJS_DIR)/llib-lunix.ln
  55 GEN_LINT_LIB     = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)/llib-lgenunix.ln

  57 #
  58 #       Include the makefiles which define build rule templates, the
```



```
  59 #       collection of files per module, and a few specific flags. Note
  60 #       that order is significant, just as with an include path. The
  61 #       first build rule template which matches the files name will be
  62 #       used. By including these in order from most machine dependent
  63 #       to most machine independent, we allow a machine dependent file
  64 #       to be used in preference over a machine independent version
  65 #       (Such as a machine specific optimization, which preserves the
  66 #       interfaces.)
  67 #
  68 include $(UTSTREE)/intel/Makefile.files
  69 include $(UTSTREE)/common/Makefile.files

  71 #
  72 # ----- TRANSITIONAL SECTION --------------------------------------------------
  73 #

  75 #
  76 #       Not everything which *should* be a module is a module yet. The
  77 #       following is a list of such objects which are currently part of
  78 #       genunix but which might someday become kmods.  This must be
  79 #       defined before we include Makefile.uts, or else genunix's build
  80 #       won't be as parallel as we might like.
  81 #
  82 NOT_YET_KMODS    = $(OLDPTY_OBJS) $(PTY_OBJS) $(VCONS_CONF_OBJS) $(MOD_OBJS)

  84 #
  85 # ----- END OF TRANSITIONAL SECTION -------------------------------------------
  86 #
  87 #       Include machine independent rules. Note that this does not imply
  88 #       that the resulting module from rules in Makefile.uts is machine
  89 #       independent. Only that the build rules are machine independent.
  90 #
  91 include $(UTSBASE)/Makefile.uts

  93 #
  94 #       The following must be defined for all implementations:
  95 #
  96 MODSTUBS               = $(UTSBASE)/intel/ia32/ml/modstubs.s

  98 #
  99 #       Define supported builds
 100 #
 101 DEF_BUILDS             = $(DEF_BUILDS64) $(DEF_BUILDS32)
 102 ALL_BUILDS             = $(ALL_BUILDS64) $(ALL_BUILDS32)

 104 #
 105 #       x86 or amd64 inline templates
 106 #
 107 INLINES_32             = $(UTSBASE)/intel/ia32/ml/ia32.il
 108 INLINES_64             = $(UTSBASE)/intel/amd64/ml/amd64.il
 109 INLINES                += $(INLINES_$(CLASS))

 111 #
 112 #       kernel-specific optimizations; override default in Makefile.master
 113 #

 115 CFLAGS_XARCH_32        = $(i386_CFLAGS)
 116 CFLAGS_XARCH_64        = $(amd64_CFLAGS)
 117 CFLAGS_XARCH           = $(CFLAGS_XARCH_$(CLASS))

 119 COPTFLAG_32            = $(COPTFLAG)
 120 COPTFLAG_64            = $(COPTFLAG64)
 121 COPTIMIZE              = $(COPTFLAG_$(CLASS))

 123 CFLAGS                 = $(CFLAGS_XARCH)
 124 CFLAGS                 += $(COPTIMIZE)
```

```
125 CFLAGS                  += $(INLINES) -D_ASM_INLINES
126 CFLAGS                  += $(CCMODE)
127 CFLAGS                  += $(SPACEFLAG)
128 CFLAGS                  += $(CCUNBOUND)
129 CFLAGS                  += $(CFLAGS_uts)
130 CFLAGS                  += -xstrconst

132 ASFLAGS_XARCH_32        = $(i386_ASFLAGS)
133 ASFLAGS_XARCH_64        = $(amd64_ASFLAGS)
134 ASFLAGS_XARCH           = $(ASFLAGS_XARCH_$(CLASS))

136 ASFLAGS                 += $(ASFLAGS_XARCH)

138 #
139 #       Define the base directory for installation.
140 #
141 BASE_INS_DIR    = $(ROOT)

143 #
144 #       Debugging level
145 #
146 #       Special knowledge of which special debugging options affect which
147 #       file is used to optimize the build if these flags are changed.
148 #
149 DEBUG_DEFS_OBJ32        =
150 DEBUG_DEFS_DBG32        = -DDEBUG
151 DEBUG_DEFS_OBJ64        =
152 DEBUG_DEFS_DBG64        = -DDEBUG
153 DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

155 DEBUG_COND_OBJ32        = $(POUND_SIGN)
156 DEBUG_COND_DBG32        =
157 DEBUG_COND_OBJ64        = $(POUND_SIGN)
158 DEBUG_COND_DBG64        =
159 IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

161 $(IF_DEBUG_OBJ)syscall.o       :=      DEBUG_DEFS      += -DSYSCALLTRACE
162 $(IF_DEBUG_OBJ)clock.o         :=      DEBUG_DEFS      += -DKSLICE=1

164 #
165 #       Collect the preprocessor definitions to be associated with *all*
166 #       files.
167 #
168 ALL_DEFS         = $(DEBUG_DEFS) $(OPTION_DEFS)

170 #
171 #       The kernels modules which are "implementation architecture"
172 #       specific for this machine are enumerated below. Note that most
173 #       of these modules must exist (in one form or another) for each
174 #       architecture.
175 #
176 #       Common Drivers (usually pseudo drivers) (/kernel/drv)
177 #       DRV_KMODS are built both 32-bit and 64-bit
178 #       DRV_KMODS_32 are built only 32-bit
179 #       DRV_KMODS_64 are built only 64-bit
180 #
181 DRV_KMODS       += aac
182 DRV_KMODS       += aggr
183 DRV_KMODS       += ahci
184 DRV_KMODS       += amd64_gart
185 DRV_KMODS       += amr
186 DRV_KMODS       += agpgart
187 DRV_KMODS       += srn
188 DRV_KMODS       += agptarget
189 DRV_KMODS       += arn
190 DRV_KMODS       += arp
```



```
191 DRV_KMODS       += asy
192 DRV_KMODS       += ata
193 DRV_KMODS       += ath
194 DRV_KMODS       += atu
195 DRV_KMODS       += audio
196 DRV_KMODS       += audio1575
197 DRV_KMODS       += audio810
198 DRV_KMODS       += audiocmi
199 DRV_KMODS       += audiocmihd
200 DRV_KMODS       += audioemu10k
201 DRV_KMODS       += audioens
202 DRV_KMODS       += audiohd
203 DRV_KMODS       += audioixp
204 DRV_KMODS       += audiols
205 DRV_KMODS       += audiop16x
206 DRV_KMODS       += audiopci
207 DRV_KMODS       += audiosolo
208 DRV_KMODS       += audiots
209 DRV_KMODS       += audiovia823x
210 DRV_KMODS_32    += audiovia97
211 DRV_KMODS       += bl
212 DRV_KMODS       += blkdev
213 DRV_KMODS       += bge
214 DRV_KMODS       += bofi
215 DRV_KMODS       += bpf
216 DRV_KMODS       += bridge
217 DRV_KMODS       += bscbus
218 DRV_KMODS       += bscv
219 DRV_KMODS       += chxge
220 DRV_KMODS       += cxgbe
221 DRV_KMODS       += ntxn
222 DRV_KMODS       += myri10ge
223 DRV_KMODS       += clone
224 DRV_KMODS       += cmdk
225 DRV_KMODS       += cn
226 DRV_KMODS       += conskbd
227 DRV_KMODS       += consms
228 DRV_KMODS       += cpqary3
229 DRV_KMODS       += cpuid
230 DRV_KMODS       += cpunex
231 DRV_KMODS       += crypto
232 DRV_KMODS       += cryptoadm
233 DRV_KMODS       += dca
234 DRV_KMODS       += devinfo
235 DRV_KMODS       += dld
236 DRV_KMODS       += dlpistub
237 DRV_KMODS_32    += dnet
238 DRV_KMODS       += dump
239 DRV_KMODS       += ecpp
240 DRV_KMODS       += emlxs
241 DRV_KMODS       += fd
242 DRV_KMODS       += fdc
243 DRV_KMODS       += fm
244 DRV_KMODS       += fssnap
245 DRV_KMODS       += hxge
246 DRV_KMODS       += i8042
247 DRV_KMODS       += i915
248 DRV_KMODS       += icmp
249 DRV_KMODS       += icmp6
250 DRV_KMODS       += intel_nb5000
251 DRV_KMODS       += intel_nhm
252 DRV_KMODS       += ip
253 DRV_KMODS       += ip6
254 DRV_KMODS       += ipf
255 DRV_KMODS       += ipnet
256 DRV_KMODS       += ippctl
```

```
257 DRV_KMODS       += ipsecah
258 DRV_KMODS       += ipsecesp
259 DRV_KMODS       += ipw
260 DRV_KMODS       += iwh
261 DRV_KMODS       += iwi
262 DRV_KMODS       += iwk
263 DRV_KMODS       += iwp
264 DRV_KMODS       += iwscn
265 DRV_KMODS       += kb8042
266 DRV_KMODS       += keysock
267 DRV_KMODS       += kssl
268 DRV_KMODS       += kstat
269 DRV_KMODS       += ksyms
270 DRV_KMODS       += kmdb
271 DRV_KMODS       += llc1
272 DRV_KMODS       += lofi
273 DRV_KMODS       += log
274 DRV_KMODS       += logindmux
275 DRV_KMODS       += mega_sas
276 DRV_KMODS       += mc-amd
277 DRV_KMODS       += mm
278 DRV_KMODS       += mouse8042
279 DRV_KMODS       += mpt_sas
280 DRV_KMODS       += mr_sas
281 DRV_KMODS       += mwl
282 DRV_KMODS       += nca
283 DRV_KMODS       += nsmb
284 DRV_KMODS       += nulldriver
285 DRV_KMODS       += nv_sata
286 DRV_KMODS       += nxge
287 DRV_KMODS       += oce
288 DRV_KMODS       += openeepr
289 DRV_KMODS       += pci_pci
290 DRV_KMODS       += pcic
291 DRV_KMODS       += pcieb
292 DRV_KMODS       += physmem
293 DRV_KMODS       += pit_beep
294 DRV_KMODS       += pm
295 DRV_KMODS       += poll
296 DRV_KMODS       += pool
297 DRV_KMODS       += power
298 DRV_KMODS       += pseudo
299 DRV_KMODS       += ptc
300 DRV_KMODS       += ptm
301 DRV_KMODS       += pts
302 DRV_KMODS       += ptsl
303 DRV_KMODS       += qlge
304 DRV_KMODS       += radeon
305 DRV_KMODS       += ral
306 DRV_KMODS       += ramdisk
307 DRV_KMODS       += random
308 DRV_KMODS       += rds
309 DRV_KMODS       += rdsv3
310 DRV_KMODS       += rpcib
311 DRV_KMODS       += rsm
312 DRV_KMODS       += rts
313 DRV_KMODS       += rtw
314 DRV_KMODS       += rum
315 DRV_KMODS       += rwd
316 DRV_KMODS       += rwn
317 DRV_KMODS       += sad
318 DRV_KMODS       += sd
319 DRV_KMODS       += sdhost
320 DRV_KMODS       += sgen
321 DRV_KMODS       += si3124
322 DRV_KMODS       += smbios
```



```
323 DRV_KMODS       += softmac
324 DRV_KMODS       += spdsock
325 DRV_KMODS       += smbsrv
326 DRV_KMODS       += smp
327 DRV_KMODS       += sppp
328 DRV_KMODS       += sppptun
329 DRV_KMODS       += srpt
330 DRV_KMODS       += st
331 DRV_KMODS       += sy
332 DRV_KMODS       += sysevent
333 DRV_KMODS       += sysmsg
334 DRV_KMODS       += tcp
335 DRV_KMODS       += tcp6
336 DRV_KMODS       += tl
337 DRV_KMODS       += tnf
338 DRV_KMODS       += tpm
339 DRV_KMODS       += trill
340 DRV_KMODS       += udp
341 DRV_KMODS       += udp6
342 DRV_KMODS       += ucode
343 DRV_KMODS       += ural
344 DRV_KMODS       += uath
345 DRV_KMODS       += urtw
346 DRV_KMODS       += vgatext
347 DRV_KMODS       += heci
348 DRV_KMODS       += vnic
349 DRV_KMODS       += vscan
350 DRV_KMODS       += wc
351 DRV_KMODS       += winlock
352 DRV_KMODS       += wpi
353 DRV_KMODS       += xge
354 DRV_KMODS       += yge
355 DRV_KMODS       += zcons
356 DRV_KMODS       += zyd
357 DRV_KMODS       += simnet
358 DRV_KMODS       += stmf
359 DRV_KMODS       += stmf_sbd
360 DRV_KMODS       += fct
361 DRV_KMODS       += fcoe
362 DRV_KMODS       += fcoet
363 DRV_KMODS       += fcoei
364 DRV_KMODS       += qlt
365 DRV_KMODS       += iscsit
366 DRV_KMODS       += pppt
367 DRV_KMODS       += ncall nsctl sdbc nskern sv
368 DRV_KMODS       += ii rdc rdcsrv rdcstub
369 DRV_KMODS       += iptun

371 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += bmc
372 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += glm
373 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += intel_nhmex
374 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += cpqary3
375 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += marvell88sx
376 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += bcm_sata
377 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += memtest
378 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += mpt
379 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += atiatom
380 $(CLOSED_BUILD)CLOSED_DRV_KMODS         += acpi_toshiba

371 #
372 # Common code drivers
373 #

375 DRV_KMODS       += afe
376 DRV_KMODS       += atge
377 DRV_KMODS       += bfe
```

```
DRV_KMODS       += dmfe
DRV_KMODS       += e1000g
DRV_KMODS       += efe
DRV_KMODS       += elxl
DRV_KMODS       += hme
DRV_KMODS       += mxfe
DRV_KMODS       += nge
DRV_KMODS       += pcn
DRV_KMODS       += rge
DRV_KMODS       += rtls
DRV_KMODS       += sfe
DRV_KMODS       += amd8111s
DRV_KMODS       += igb
DRV_KMODS       += ipmi
DRV_KMODS       += iprb
DRV_KMODS       += ixgbe
DRV_KMODS       += vr
$(CLOSED_BUILD)CLOSED_DRV_KMODS += ixgb

#
# Virtio drivers
#

# Virtio core
DRV_KMODS       += virtio

# Virtio block driver
DRV_KMODS       += vioblk

#
#       DTrace and DTrace Providers
#
DRV_KMODS       += dtrace
DRV_KMODS       += fbt
DRV_KMODS       += lockstat
DRV_KMODS       += profile
DRV_KMODS       += sdt
DRV_KMODS       += systrace
DRV_KMODS       += fasttrap
DRV_KMODS       += dcpc

#
#       I/O framework test drivers
#
DRV_KMODS       += pshot
DRV_KMODS       += gen_drv
DRV_KMODS       += tvhci tphci tclient
DRV_KMODS       += emul64

#
#       Machine Specific Driver Modules (/kernel/drv):
#
DRV_KMODS       += options
DRV_KMODS       += scsi_vhci
DRV_KMODS       += pmcs
DRV_KMODS       += pmcs8001fw
DRV_KMODS       += arcmsr
DRV_KMODS       += fcp
DRV_KMODS       += fcip
DRV_KMODS       += fcsm
DRV_KMODS       += fp
DRV_KMODS       += qlc
DRV_KMODS       += iscsi

#
#       PCMCIA specific module(s)
```



```
#
DRV_KMODS       += pcs
MISC_KMODS      += cardbus

#
#       SCSI Enclosure Services driver
#
DRV_KMODS       += ses

#
#       USB specific modules
#
DRV_KMODS       += hid
DRV_KMODS       += hwarc hwahc
DRV_KMODS       += hubd
DRV_KMODS       += uhci
DRV_KMODS       += ehci
DRV_KMODS       += ohci
DRV_KMODS       += usb_mid
DRV_KMODS       += usb_ia
DRV_KMODS       += scsa2usb
DRV_KMODS       += usbprn
DRV_KMODS       += ugen
DRV_KMODS       += usbser
DRV_KMODS       += usbsacm
DRV_KMODS       += usbsksp
DRV_KMODS       += usbsprl
DRV_KMODS       += usb_ac
DRV_KMODS       += usb_as
DRV_KMODS       += usbskel
DRV_KMODS       += usbvc
DRV_KMODS       += usbftdi
DRV_KMODS       += wusb_df
DRV_KMODS       += wusb_ca
DRV_KMODS       += usbecm

$(CLOSED_BUILD)CLOSED_DRV_KMODS += usbser_edge

#
#       1394 modules
#
MISC_KMODS      += s1394 sbp2
DRV_KMODS       += hci1394 scsa1394
DRV_KMODS       += av1394
DRV_KMODS       += dcam1394

#
#       InfiniBand pseudo drivers
#
DRV_KMODS       += ib ibp eibnx eoib rdsib sdp iser daplt hermon tavor sol_ucma
DRV_KMODS       += sol_umad

#
#       LVM modules
#
DRV_KMODS       += md
MISC_KMODS      += md_stripe md_hotspares md_mirror md_raid md_trans md_notify
MISC_KMODS      += md_sp

#
#       Brand modules
#
BRAND_KMODS     += sn1_brand s10_brand

#
#       Exec Class Modules (/kernel/exec):
```

```
507 #
508 EXEC_KMODS      += elfexec intpexec shbinexec javaexec

510 #
511 #       Scheduling Class Modules (/kernel/sched):
512 #
513 SCHED_KMODS     += IA RT TS RT_DPTBL TS_DPTBL FSS FX FX_DPTBL SDC

515 #
516 #       File System Modules (/kernel/fs):
517 #
518 FS_KMODS        += autofs cachefs ctfs dcfs dev devfs fdfs fifofs hsfs lofs
519 FS_KMODS        += mntfs namefs nfs objfs zfs zut
520 FS_KMODS        += pcfs procfs sockfs specfs tmpfs udfs ufs sharefs
521 FS_KMODS        += smbfs

523 #
524 #       Streams Modules (/kernel/strmod):
525 #
526 STRMOD_KMODS    += bufmod connld dedump ldterm pckt pfmod pipemod
527 STRMOD_KMODS    += ptem redirmod rpcmod rlmod telmod timod
528 STRMOD_KMODS    += spppasyn spppcomp
529 STRMOD_KMODS    += tirdwr ttcompat
530 STRMOD_KMODS    += usbkbm
531 STRMOD_KMODS    += usbms
532 STRMOD_KMODS    += usbwcm
533 STRMOD_KMODS    += usb_ah
534 STRMOD_KMODS    += drcompat
535 STRMOD_KMODS    += cryptmod
536 STRMOD_KMODS    += vuid2ps2
537 STRMOD_KMODS    += vuid3ps2
538 STRMOD_KMODS    += vuidm3p
539 STRMOD_KMODS    += vuidm4p
540 STRMOD_KMODS    += vuidm5p

542 #
543 #       'System' Modules (/kernel/sys):
544 #
545 SYS_KMODS       += c2audit
546 SYS_KMODS       += doorfs
547 SYS_KMODS       += exacctsys
548 SYS_KMODS       += inst_sync
549 SYS_KMODS       += kaio
550 SYS_KMODS       += msgsys
551 SYS_KMODS       += pipe
552 SYS_KMODS       += portfs
553 SYS_KMODS       += pset
554 SYS_KMODS       += semsys
555 SYS_KMODS       += shmsys
556 SYS_KMODS       += sysacct
557 SYS_KMODS       += acctctl

559 #
560 #       'Misc' Modules (/kernel/misc)
561 #       MISC_KMODS are built both 32-bit and 64-bit
562 #       MISC_KMODS_32 are built only 32-bit
563 #       MISC_KMODS_64 are built only 64-bit
564 #
565 MISC_KMODS      += ac97
566 MISC_KMODS      += acpica
567 MISC_KMODS      += agpmaster
568 MISC_KMODS      += bignum
569 MISC_KMODS      += bootdev
570 MISC_KMODS      += busra
571 MISC_KMODS      += cmlb
572 MISC_KMODS      += consconfig
```



```
573 MISC_KMODS      += ctf
574 MISC_KMODS      += dadk
575 MISC_KMODS      += dcopy
576 MISC_KMODS      += dls
577 MISC_KMODS      += drm
578 MISC_KMODS      += fssnap_if
579 MISC_KMODS      += gda
580 MISC_KMODS      += gld
581 MISC_KMODS      += hidparser
582 MISC_KMODS      += hook
583 MISC_KMODS      += hpcsvc
584 MISC_KMODS      += ibcm
585 MISC_KMODS      += ibdm
586 MISC_KMODS      += ibdma
587 MISC_KMODS      += ibmf
588 MISC_KMODS      += ibtl
589 MISC_KMODS      += idm
590 MISC_KMODS      += idmap
591 MISC_KMODS      += iommulib
592 MISC_KMODS      += ipc
593 MISC_KMODS      += kbtrans
594 MISC_KMODS      += kcf
595 MISC_KMODS      += kgssapi
596 MISC_KMODS      += kmech_dummy
597 MISC_KMODS      += kmech_krb5
598 MISC_KMODS      += ksocket
599 MISC_KMODS      += mac
600 MISC_KMODS      += mii
601 MISC_KMODS      += mwlfw
602 MISC_KMODS      += net80211
603 MISC_KMODS      += nfs_dlboot
604 MISC_KMODS      += nfssrv
605 MISC_KMODS      += neti
606 MISC_KMODS      += pci_autoconfig
607 MISC_KMODS      += pcicfg
608 MISC_KMODS      += pcihp
609 MISC_KMODS      += pcmcia
610 MISC_KMODS      += rpcsec
611 MISC_KMODS      += rpcsec_gss
612 MISC_KMODS      += rsmops
613 MISC_KMODS      += sata
614 MISC_KMODS      += scsi
615 MISC_KMODS      += sda
616 MISC_KMODS      += sol_ofs
617 MISC_KMODS      += spuni
618 MISC_KMODS      += strategy
619 MISC_KMODS      += strplumb
620 MISC_KMODS      += tem
621 MISC_KMODS      += tlimod
622 MISC_KMODS      += usba usba10 usbs49_fw
623 MISC_KMODS      += scsi_vhci_f_sym_hds
624 MISC_KMODS      += scsi_vhci_f_sym
625 MISC_KMODS      += scsi_vhci_f_tpgs
626 MISC_KMODS      += scsi_vhci_f_asym_sun
627 MISC_KMODS      += scsi_vhci_f_tape
628 MISC_KMODS      += scsi_vhci_f_tpgs_tape
629 MISC_KMODS      += fctl
630 MISC_KMODS      += emlxs_fw
631 MISC_KMODS      += qlc_fw_2200
632 MISC_KMODS      += qlc_fw_2300
633 MISC_KMODS      += qlc_fw_2400
634 MISC_KMODS      += qlc_fw_2500
635 MISC_KMODS      += qlc_fw_6322
636 MISC_KMODS      += qlc_fw_8100
637 MISC_KMODS      += hwa1480_fw
638 MISC_KMODS      += uathfw
```

```
MISC_KMODS      += uwba

MISC_KMODS      += klmmod klmops

$(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_asym_lsi
$(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_asym_emc
$(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_sym_emc

#
#       Software Cryptographic Providers (/kernel/crypto):
#
CRYPTO_KMODS    += aes
CRYPTO_KMODS    += arcfour
CRYPTO_KMODS    += blowfish
CRYPTO_KMODS    += des
CRYPTO_KMODS    += ecc
CRYPTO_KMODS    += md4
CRYPTO_KMODS    += md5
CRYPTO_KMODS    += rsa
CRYPTO_KMODS    += sha1
CRYPTO_KMODS    += sha2
CRYPTO_KMODS    += swrand

#
#       IP Policy Modules (/kernel/ipp)
#
IPP_KMODS       += dlcosmk
IPP_KMODS       += flowacct
IPP_KMODS       += ipgpc
IPP_KMODS       += dscpmk
IPP_KMODS       += tokenmt
IPP_KMODS       += tswtclmt

#
#       generic-unix module (/kernel/genunix):
#
GENUNIX_KMODS   += genunix

#
#       SVVS Testing Modules (/kernel/strmod):
#
#       These are streams and driver modules which are not to be
#       delivered with a released system. However, during development
#       it is convenient to build and install the SVVS kernel modules.
#
SVVS_KMODS      += lmodb lmode lmodr lmodt svvslo tidg tivc tmux

$(CLOSED_BUILD)SVVS             += svvs

#
#       Modules eXcluded from the product:
#
$(CLOSED_BUILD)CLOSED_XMODS =           \
        adpu320         \
        bnx             \
        bnxe            \
        lsimega         \
        sdpib


#
#       'Dacf' Modules (/kernel/dacf):
#

#
#       Performance Counter BackEnd modules (/usr/kernel/pcbe)
```



```
#
PCBE_KMODS      += p123_pcbe p4_pcbe opteron_pcbe core_pcbe

#
#       MAC-Type Plugin Modules (/kernel/mac)
#
MAC_KMODS       += mac_6to4
MAC_KMODS       += mac_ether
MAC_KMODS       += mac_ipv4
MAC_KMODS       += mac_ipv6
MAC_KMODS       += mac_wifi
MAC_KMODS       += mac_ib

#
# socketmod (kernel/socketmod)
#
SOCKET_KMODS    += sockpfp
SOCKET_KMODS    += socksctp
SOCKET_KMODS    += socksdp
SOCKET_KMODS    += sockrds
SOCKET_KMODS    += ksslf

#
#       kiconv modules (/kernel/kiconv):
#
KICONV_KMODS    += kiconv_emea kiconv_ja kiconv_ko kiconv_sc kiconv_tc

#
#       'Dacf' Modules (/kernel/dacf):
#
DACF_KMODS      += net_dacf
```

```
**********************************************************
    2185 Wed Oct 16 17:41:09 2013
new/usr/src/uts/intel/intel_nb5000/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 #
  20 # Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
  21 # Use is subject to license terms.
  22 # Copyright (c) 2011 Bayard G. Bell. All rights reserved.
  23 #

  25 #
  26 # Intel 5000/5100/5400/7300 chipset memory controller hub (MCH) module
  27 #

  29 #
  30 #       Path to the base of the uts directory tree (usually /usr/src/uts).
  31 #
  32 UTSBASE = ../..
  33 UTSCLOSED = ../../../../closed/uts

  34 #
  35 #       Define the module and object file sets.
  36 #
  37 MODULE          = intel_nb5000
  38 #
  39 ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
  40 SRCDIR          = $(UTSBASE)/intel/io/intel_nb5000
  41 CONF_SRCDIR     = $(UTSBASE)/intel/io/intel_nb5000

  43 #
  44 #       Include common rules.
  45 #
  46 include ../Makefile.intel

  49 #
  50 # The list of object files is defined here, rather than in Makefile.files,
  51 # because the "$(CLOSED_BUILD)" macro has not been defined at the time
  52 # Makefile.files is processed.
  53 #
  54 INTEL_NB5000_OBJS += \
  55         intel_nb5000.o \
  56         intel_nbdrv.o \
  57         dimm_addr.o \
  58         nb_pci_cfg.o \
```



```
  59         nb5000_init.o

  61 $(CLOSED_BUILD)INTEL_NB5000_OBJS += memtrans.o

  48 OBJECTS         = $(INTEL_NB5000_OBJS:%=$(OBJS_DIR)/%)
  49 LINTS           = $(INTEL_NB5000_OBJS:%.o=$(LINTS_DIR)/%.ln)

  51 #
  52 #       Define targets
  53 #
  54 ALL_TARGET      = $(BINARY)
  55 LINT_TARGET     = $(LINT_MODULE).lint
  56 INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

  58 #
  59 # Depends on smbios
  60 #
  61 LDFLAGS         += -dy -N drv/smbios

  63 CERRWARN        += -_gcc=-Wno-parentheses
  64 CERRWARN        += -_gcc=-Wno-uninitialized

  66 #
  67 #       Default build targets.
  68 #
  69 .KEEP_STATE:

  71 def:            $(DEF_DEPS)

  73 all:            $(ALL_DEPS)

  75 clean:          $(CLEAN_DEPS)

  77 clobber:        $(CLOBBER_DEPS)

  79 lint:           $(LINT_DEPS)

  81 modlintlib:     $(MODLINTLIB_DEPS)

  83 clean.lint:     $(CLEAN_LINT_DEPS)

  85 install:        $(INSTALL_DEPS)

  87 $(OBJECTS): $(OBJS_DIR) $(MCAMD_OFF_H)

  89 #
  90 #       Include common targets.
  91 #
  92 include ../Makefile.targ
 108 $(CLOSED_BUILD)include $(UTSCLOSED)/intel/Makefile.rules
```

```
**********************************************************
    2012 Wed Oct 16 17:41:09 2013
new/usr/src/uts/intel/intel_nhm/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 #
  20 # Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
  21 # Use is subject to license terms.
  22 #

  24 #
  25 # Intel Nehalem memory controller module
  26 #

  28 #
  29 #       Path to the base of the uts directory tree (usually /usr/src/uts).
  30 #
  31 UTSBASE = ../..
  32 UTSCLOSED = ../../../../closed/uts

  33 #
  34 #       Define the module and object file sets.
  35 #
  36 MODULE          = intel_nhm
  37 #
  38 ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
  39 CONF_SRCDIR     = $(UTSBASE)/intel/io/intel_nhm

  41 #
  42 #       Include common rules.
  43 #
  44 include $(UTSBASE)/intel/Makefile.intel

  47 #
  48 # The list of object files is defined here, rather than in Makefile.files,
  49 # because the "$(CLOSED_BUILD)" macro has not been defined at the time
  50 # Makefile.files is processed.
  51 #
  52 INTEL_NHM_OBJS += \
  53         nhm_init.o \
  54         mem_addr.o \
  55         intel_nhmdrv.o \
  56         nhm_pci_cfg.o \
  57         dimm_topo.o \
  58         intel_nhm.o
```



```
  60 $(CLOSED_BUILD)INTEL_NHM_OBJS += \
  61         nhm_dimm_addr.o
```

```
  46 OBJECTS         = $(INTEL_NHM_OBJS:%=$(OBJS_DIR)/%)
  47 LINTS           = $(INTEL_NHM_OBJS:%.o=$(LINTS_DIR)/%.ln)

  49 #
  50 #       Define targets
  51 #
  52 ALL_TARGET      = $(BINARY) $(SRC_CONFFILE)
  53 LINT_TARGET     = $(LINT_MODULE).lint
  54 INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

  56 CPPFLAGS        +=  -I$(UTSBASE)/i86pc
  57 LDFLAGS         += -dy -N drv/smbios

  59 CERRWARN        += -_gcc=-Wno-uninitialized

  61 #
  62 #       Default build targets.
  63 #
  64 .KEEP_STATE:

  66 def:             $(DEF_DEPS)

  68 all:             $(ALL_DEPS)

  70 clean:           $(CLEAN_DEPS)

  72 clobber:         $(CLOBBER_DEPS)

  74 lint:            $(LINT_DEPS)

  76 modlintlib:      $(MODLINTLIB_DEPS)

  78 clean.lint:      $(CLEAN_LINT_DEPS)

  80 install:         $(INSTALL_DEPS)

  82 #
  83 #       Include common targets.
  84 #
  85 include ../Makefile.targ
 103 $(CLOSED_BUILD)include $(UTSCLOSED)/intel/Makefile.rules
```

```
**********************************************************
    3889 Wed Oct 16 17:41:10 2013
new/usr/src/uts/intel/sd/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# uts/intel/io/sd/Makefile
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
#
#       This makefile drives the production of the sd
#       kernel module.
#
#


#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = sd
OBJECTS         = $(SD_OBJS:%=$(OBJS_DIR)/%)
LINTS           = $(SD_OBJS:%.o=$(LINTS_DIR)/%.ln)
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
CONF_SRCDIR     = $(UTSBASE)/intel/io/scsi/targets
WARLOCK_OUT     = $(SD_OBJS:%.o=%.ll)
WARLOCK_OK      = $(MODULE).ok
WLCMD_DIR       = $(UTSBASE)/common/io/warlock

#
#       Include common rules.
#
include $(UTSBASE)/intel/Makefile.intel

#
#       Define targets
#
ALL_TARGET      = $(BINARY) $(CONFMOD)
LINT_TARGET     = $(MODULE).lint
```



```
INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

#
#       Overrides.
#
DEBUG_FLGS      =
DEBUG_DEFS      += $(DEBUG_FLGS)

INC_PATH        += -I$(UTSBASE)/intel/io/scsi/targets

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_STATIC_UNUSED
LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV

CERRWARN        += -_gcc=-Wno-unused-variable
CERRWARN        += -_gcc=-Wno-unused-function
CERRWARN        += -_gcc=-Wno-unused-label
CERRWARN        += -_gcc=-Wno-parentheses
CERRWARN        += -_gcc=-Wno-type-limits
CERRWARN        += -_gcc=-Wno-uninitialized

#
# Depends on scsi and cmlb
#
LDFLAGS         += -dy -N misc/scsi -N misc/cmlb

#
#       Default build targets.
#
.KEEP_STATE:

def:            $(DEF_DEPS)

all:            $(ALL_DEPS)

clean:          $(CLEAN_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

clobber:        $(CLOBBER_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

lint:           $(LINT_DEPS)

modlintlib:     $(MODLINTLIB_DEPS)

clean.lint:     $(CLEAN_LINT_DEPS)

install:        $(INSTALL_DEPS)

#
#       Include common targets.
#
include $(UTSBASE)/intel/Makefile.targ


#
#       Defines for local commands.
#
WARLOCK         = warlock
```

```
125 WLCC            = wlcc
126 TOUCH           = touch
127 TEST            = test

129 #
130 # Warlock targets
131 #
132 # Note that in warlock_with_{esp,isp} it is important to load sd.ll
133 # before {isp,esp}.ll; the reason is that both have _init/_info/_fini
134 # and warlock can only handle one extern function by a given name;
135 # any loaded after the first are ignored.

137 SCSI_FILES = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)
138 CMLB_FILES = $(CMLB_OBJS:%.o=-l ../cmlb/%.ll)

140 WARLOCK_TARGETS = warlock_alone
141 $(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_mpt warlock_with_glm

142 warlock: $(WARLOCK_TARGETS)

144 warlock_alone: $(WARLOCK_OK)

146 $(WARLOCK_OK):  $(WLCMD_DIR)/sd.wlcmd $(WARLOCK_OUT) scsi_files \
147         warlock_ddi.files cmlb_files
148         $(WARLOCK) -c $(WLCMD_DIR)/sd.wlcmd $(WARLOCK_OUT) $(SCSI_FILES) \
149                 $(CMLB_FILES) \
150                 -l ../warlock/ddi_dki_impl.ll
151         $(TOUCH) $@

153 %.ll: $(UTSBASE)/common/io/scsi/targets/%.c
154         $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

157 warlock_with_glm: $(WLCMD_DIR)/sd_with_glm.wlcmd $(WARLOCK_OUT) scsi_files \
158         glm_files warlock_ddi.files cmlb_files
159         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_glm.wlcmd \
160                 $(WARLOCK_OUT) \
161                 $(CLOSED)/uts/intel/glm/*.ll \
162                  $(SCSI_FILES) $(CMLB_FILES) \
163                 -l ../warlock/ddi_dki_impl.ll

165 warlock_with_mpt: $(WLCMD_DIR)/sd_with_mpt.wlcmd $(WARLOCK_OUT) scsi_files \
166         mpt_files warlock_ddi.files cmlb_files
167         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_mpt.wlcmd \
168                 $(WARLOCK_OUT) \
169                 $(CLOSED)/uts/intel/mpt/*.ll \
170                 $(SCSI_FILES) $(CMLB_FILES) \
171                 -l ../warlock/ddi_dki_impl.ll

173 glm_files:
174         @cd $(CLOSED)/uts/intel/glm; pwd; $(MAKE) warlock

176 mpt_files:
177         @cd $(CLOSED)/uts/intel/mpt; pwd; $(MAKE) warlock

156 cmlb_files:
157         @cd ../cmlb; pwd; $(MAKE) warlock

159 scsi_files:
160         @cd ../scsi; pwd; $(MAKE) warlock

162 warlock_ddi.files:
163         @cd ../warlock; pwd; $(MAKE) warlock
```

```
**********************************************************
    3198 Wed Oct 16 17:41:10 2013
new/usr/src/uts/intel/ses/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# uts/intel/ses/Makefile
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
#
#       This makefile drives the production of the ses (SCSI enclosure
#       services) driver kernel module.
#
#       intel implementation architecture dependent
#


#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = ses
OBJECTS         = $(SES_OBJS:%=$(OBJS_DIR)/%)
LINTS           = $(SES_OBJS:%.o=$(LINTS_DIR)/%.ln)
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
CONF_SRCDIR     = $(UTSBASE)/intel/io/scsi/targets
WARLOCK_OUT     = $(SES_OBJS:%.o=%.ll)
WARLOCK_OK      = $(MODULE).ok
WLCMD_DIR       = $(UTSBASE)/common/io/warlock

#
#       Include common rules.
#
include $(UTSBASE)/intel/Makefile.intel

#
#       Define targets
#
```



```
ALL_TARGET      = $(BINARY) $(SRC_CONFILE)
LINT_TARGET     = $(MODULE).lint
INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)
CLEANFILES      += $(WARLOCK_TARGETS)

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN

#
# Depends on scsi
#
LDFLAGS         += -dy -N misc/scsi

#
#       Default build targets.
#
.KEEP_STATE:

def:            $(DEF_DEPS)

all:            $(ALL_DEPS)

clean:          $(CLEAN_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

clobber:        $(CLOBBER_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

lint:           $(LINT_DEPS)

modlintlib:     $(MODLINTLIB_DEPS)

clean.lint:     $(CLEAN_LINT_DEPS)

install:        $(INSTALL_DEPS)

#
#       Include common targets.
#
include $(UTSBASE)/intel/Makefile.targ


#
# Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test

#
# Warlock targets
#
SCSI_FILES      = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)

WARLOCK_TARGETS = $(WARLOCK_OK)
$(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_glm

warlock: $(WARLOCK_TARGETS)

# XXX FIX ME: why only ses.ll?
```

```
125 $(WARLOCK_OK): $(WLCMD_DIR)/$(MODULE).wlcmd $(WARLOCK_OUT)
126         @cd ../warlock; $(MAKE) warlock
127         @cd ../scsi; $(MAKE) warlock
128         $(WARLOCK) -c $(WLCMD_DIR)/$(MODULE).wlcmd ses.ll \
129                 ../warlock/scsi.ll \
130                 -l ../warlock/ddi_dki_impl.ll  \
131                 $(SCSI_FILES)
132         @ $(TOUCH) $@

134 %.ll: $(UTSBASE)/common/io/scsi/targets/%.c
135         $(WLCC) $(CPPFLAGS) -o $@ $<

138 warlock_with_glm: $(WLCMD_DIR)/ses_with_glm.wlcmd $(WARLOCK_OUT) glm_files
139         @cd ../warlock; $(MAKE) warlock
140         @cd ../scsi; $(MAKE) warlock
141         @cd $(CLOSED)/uts/intel/glm; $(MAKE) warlock;
142         $(WARLOCK) -c $(WLCMD_DIR)/ses_with_glm.wlcmd ses.ll \
143                 $(CLOSED)/uts/intel/glm/*.ll \
144                 $(SCSI_FILES) \
145                 ../warlock/scsi.ll \
146                 -l ../warlock/ddi_dki_impl.ll

148 glm_files:
149         @cd $(CLOSED)/uts/intel/glm; pwd; $(MAKE) warlock

137 warlock_ddi.files:
138         @cd ../warlock; pwd; $(MAKE) warlock
```

```
**********************************************************
    3358 Wed Oct 16 17:41:10 2013
new/usr/src/uts/intel/st/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 # uts/intel/st/Makefile
  23 # Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
  24 # Use is subject to license terms.
  25 # Copyright (c) 2011 Bayard G. Bell. All rights reserved.
  26 #
  27 #       This makefile drives the production of the st driver
  28 #       kernel module.
  29 #
  30 #       intel implementation architecture independent
  31 #

  33 #
  34 #       Path to the base of the uts directory tree (usually /usr/src/uts).
  35 #
  36 UTSBASE = ../..

  38 #
  39 #       Define the module and object file sets.
  40 #
  41 MODULE          = st
  42 OBJECTS         = $(ST_OBJS:%=$(OBJS_DIR)/%)
  43 LINTS           = $(ST_OBJS:%.o=$(LINTS_DIR)/%.ln)
  44 ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
  45 CONF_SRCDIR     = $(UTSBASE)/intel/io/scsi/targets
  46 WARLOCK_OUT     = $(ST_OBJS:%.o=%.ll)
  47 WARLOCK_OK      = $(MODULE).ok
  48 WLCMD_DIR       = $(UTSBASE)/common/io/warlock

  50 #
  51 #       Include common rules.
  52 #
  53 include $(UTSBASE)/intel/Makefile.intel

  55 #
  56 #       Define targets
  57 #
  58 ALL_TARGET      = $(BINARY) $(SRC_CONFILE)
```



```
  59 LINT_TARGET     = $(MODULE).lint
  60 INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

  62 #
  63 #       Overrides.
  64 #
  65 DEBUG_FLGS      =
  66 DEBUG_DEFS      += $(DEBUG_FLGS)

  68 #
  69 # For now, disable these lint checks; maintainers should endeavor
  70 # to investigate and remove these for maximum lint coverage.
  71 # Please do not carry these forward to new Makefiles.
  72 #
  73 LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
  74 LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
  75 LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV

  77 CERRWARN        += -_gcc=-Wno-parentheses
  78 CERRWARN        += -_gcc=-Wno-switch
  79 CERRWARN        += -_gcc=-Wno-uninitialized

  81 #
  82 # Depends on scsi
  83 #
  84 LDFLAGS         += -dy -N misc/scsi

  86 #
  87 #       Default build targets.
  88 #
  89 .KEEP_STATE:

  91 def:            $(DEF_DEPS)

  93 all:            $(ALL_DEPS)

  95 clean:          $(CLEAN_DEPS)
  96                 $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

  98 clobber:        $(CLOBBER_DEPS)
  99                 $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

 101 lint:           $(LINT_DEPS)

 103 modlintlib:     $(MODLINTLIB_DEPS)

 105 clean.lint:     $(CLEAN_LINT_DEPS)

 107 install:        $(INSTALL_DEPS)

 109 #
 110 #       Include common targets.
 111 #
 112 include $(UTSBASE)/intel/Makefile.targ

 114 #
 115 #       Defines for local commands.
 116 #
 117 WARLOCK         = warlock
 118 WLCC            = wlcc
 119 TOUCH           = touch
 120 TEST            = test

 122 #
 123 # Warlock targets
 124 #
```

```
126 SCSI_FILES = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)

128 WARLOCK_TARGETS = warlock_alone
129 $(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_glm

130 warlock: $(WARLOCK_TARGETS)

132 warlock_alone: $(WARLOCK_OK)

134 scsi_files:
135         @cd ../scsi; pwd; $(MAKE) warlock

137 $(WARLOCK_OK):  $(WLCMD_DIR)/st.wlcmd $(WARLOCK_OUT) scsi_files \
138         warlock_ddi.files
139         $(WARLOCK) -c $(WLCMD_DIR)/st.wlcmd $(WARLOCK_OUT) $(SCSI_FILES) \
140         -l ../warlock/ddi_dki_impl.ll
141         $(TOUCH) $@

143 %.ll: $(UTSBASE)/common/io/scsi/targets/%.c
144         $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

147 warlock_with_glm: $(WLCMD_DIR)/st_with_glm.wlcmd $(WARLOCK_OUT) scsi_files \
148         glm_files warlock_ddi.files
149         $(WARLOCK) -c $(WLCMD_DIR)/st_with_glm.wlcmd \
150                 $(WARLOCK_OUT) $(CLOSED)/uts/intel/glm/glm $(SCSI_FILES) \
151                 -l ../warlock/ddi_dki_impl.ll
152 glm_files:
153         @cd $(CLOSED)/uts/intel/glm; pwd; $(MAKE) warlock

146 warlock_ddi.files:
147         @cd ../warlock; pwd; $(MAKE) warlock

149 scsi.files:
150         @cd ../scsi; pwd; $(MAKE) warlock
```

```
**********************************************************
    2533 Wed Oct 16 17:41:10 2013
new/usr/src/uts/intel/sys/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#

include ../../../Makefile.master

# NOTE: hrtcntl.h and hrtsys.h are present in this directory so that the
#       hrtsys system call can be built to facilitate transportability of
#       stock SVr4 programs.  Every effort is to be made to prevent objects
#       from being built, so these headers are not exported (installed).

HDRS    =                       \
        archsystm.h             \
        asm_linkage.h           \
        bootconf.h              \
        bootregs.h              \
        bootsvcs.h              \
        controlregs.h           \
        cpu.h                   \
        ddi_isa.h               \
        debugreg.h              \
        fasttrap_isa.h          \
        fp.h                    \
        frame.h                 \
        inline.h                \
        iommulib.h              \
        hypervisor.h            \
        kdi_machimpl.h          \
        kdi_regs.h              \
        machlock.h              \
        machsig.h               \
        machtypes.h             \
        mc.h                    \
        mc_amd.h                \
        mc_intel.h              \
        mca_amd.h               \
        mca_x86.h               \
```



```
        mutex_impl.h            \
        obpdefs.h               \
        old_procfs.h            \
        pcb.h                   \
        pmem.h                  \
        privmregs.h             \
        privregs.h              \
        procfs_isa.h            \
        prom_emul.h             \
        prom_isa.h              \
        prom_plat.h             \
        promif.h                \
        promimpl.h              \
        psw.h                   \
        pte.h                   \
        reg.h                   \
        regset.h                \
        segment.h               \
        segments.h              \
        spl.h                   \
        stack.h                 \
        stat_impl.h             \
        synch32.h               \
        sysconfig_impl.h        \
        sysi86.h                \
        trap.h                  \
        traptrace.h             \
        tss.h                   \
        ucontext.h              \
        utrap.h                 \
        vmparam.h               \
        x86_archext.h           \
        xen_errno.h

CLOSEDHDRS = \
        memtest.h

ROOTDIR=        $(ROOT)/usr/include/sys
SCSIDIR=        $(ROOTDIR)/scsi
SCSIDIRS=       $(SCSIDIR) $(SCSIDIR)/conf $(SCSIDIR)/generic \
                $(SCSIDIR)/impl $(SCSIDIR)/targets
ROOTFSDIR=      $(ROOTDIR)/fs
ROOTDIRS=       $(ROOTDIR) $(ROOTFSDIR)

ROOTHDRS= $(HDRS:%=$(ROOTDIR)/%)
$(CLOSED_BUILD)ROOTHDRS += $(CLOSEDHDRS:%=$(ROOTDIR)/%)

CHECKHDRS=                      \
        $(HDRS:%.h=%.check)

$(CLOSED_BUILD)CHECKHDRS += $(CLOSEDHDRS:%.h=$(CLOSED)/uts/intel/sys/%.check)

# install rules
$(ROOTDIR)/%: %
        $(INS.file)

$(ROOTDIR)/%:   $(CLOSED)/uts/intel/sys/%
        $(INS.file)

.KEEP_STATE:

.PARALLEL: $(CHECKHDRS) $(ROOTHDRS)

install_h: $(ROOTDIRS) .WAIT $(ROOTHDRS)

$(ROOTDIRS):
```

```
 116         $(INS.dir)
 118 check:  $(CHECKHDRS)
```

```
**********************************************************
    3432 Wed Oct 16 17:41:10 2013
new/usr/src/uts/intel/warlock/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright (c) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
#
# uts/intel/warlock/Makefile
#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = warlock
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)

.KEEP_STATE:

CFLAGS += -I../../common/sys -I../../sun/sys/scsi -D_KERNEL
#
#       Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test

include $(UTSBASE)/intel/Makefile.intel

#
#       lock_lint rules
#
all:    warlock warlock.1394 warlock.ecpp warlock.scsi \
        warlock.usb warlock.ib warlock.sata warlock.wc \
        warlock.audiohd

warlock: $(MODULE).ok
```



```
warlock.ok: ddi_dki_impl.ll scsi.ll
        $(TOUCH) $@

%.ll: $(UTSBASE)/common/io/warlock/%.c
        $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

warlock.usb:
        @cd ../usba; $(MAKE) clean; $(MAKE) warlock
        @cd ../ohci; $(MAKE) clean; $(MAKE) warlock
        @cd ../uhci; $(MAKE) clean; $(MAKE) warlock
        @cd ../ehci; $(MAKE) clean; $(MAKE) warlock
        @cd ../hid; $(MAKE) clean; $(MAKE) warlock
        @cd ../scsa2usb; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_ac; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_as; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_ah; $(MAKE) clean; $(MAKE) warlock
        @cd ../ugen; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_mid; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbprn; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbser; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsksp; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsprl; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsacm; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbecm; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbskel; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/usbser_edge; \
                $(MAKE) clean; $(MAKE) warlock

warlock.scsi:
        @cd ../sd; $(MAKE) clean; $(MAKE) warlock
        @cd ../ses; $(MAKE) clean; $(MAKE) warlock
        @cd ../st; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/glm; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/mpt; $(MAKE) clean; $(MAKE) warlock


warlock.1394:
        @cd ../s1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../hci1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../scsa1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../av1394; $(MAKE) clean; $(MAKE) warlock

warlock.ecpp:
        @cd ../ecpp; $(MAKE) clean; $(MAKE) warlock

warlock.ib:
        @cd ../ibmf; $(MAKE) clean; $(MAKE) warlock
        @cd ../ib; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibtl; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibcm; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibd; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/tavor; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/hermon; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/daplt; $(MAKE) clean; $(MAKE) warlock

warlock.sata:
        @cd ../sata; $(MAKE) clean; $(MAKE) warlock
        @cd ../si3124; $(MAKE) clean; $(MAKE) warlock
        @cd ../nv_sata; $(MAKE) clean; $(MAKE) warlock
        @cd ../ahci; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/intel/marvell88sx; \
                $(MAKE) clean; $(MAKE) warlock

warlock.wc:
        @cd ../wc; $(MAKE) clean; $(MAKE) warlock
```

```
115 warlock.audiohd:
116         @cd ../audiohd; $(MAKE) clean; $(MAKE) warlock
```

```
**********************************************************
    2567 Wed Oct 16 17:41:10 2013
new/usr/src/uts/sparc/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# uts/sparc/Makefile
#
#       This makefile drives the production of all implementation architecture
#       independent modules for the SPARC processor. (For those unsure, this
#       means the module will run on all SPARC processor based machines
#       running SunOS.)

UTSBASE = ..

include Makefile.sparc

LINT_KMODS_X1   = $(LINT_KMODS:nsmb=)
LINT_KMODS_X2   = $(LINT_KMODS_X1:smbfs=)
LINT_KMODLIBS   = $(LINT_KMODS_X2:e1000g=)
LINT_LIBS       = $(LINT_LIB) $(GEN_LINT_LIB) \
                  $(LINT_KMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                  $(LINT_XMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln)
                  $(LINT_KMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln)

$(CLOSED_BUILD)LINT_LIBS += $(CLOSED_LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln)

$(CLOSED_BUILD)LINT_LIBS        += $(SVVS_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln)
LINT_LIBS       += $(LINT_XMODLIBS:%=$(LINT_LIB_DIR)/llib-l%.ln)
$(CLOSED_BUILD)LINT_LIBS += $(CLOSED_XMODS:%=$(LINT_LIB_DIR)/llib-l%.ln)

DRV_KMODS       += dprov

def             :=      TARGET= def
all             :=      TARGET= all
install         :=      TARGET= install
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint
modlintlib      :=      TARGET= modlintlib
```



```
modlist         :=      TARGET= modlist
modlist         :=      NO_STATE= -K $$MODSTATE$$$$
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check
install_h       :=      TARGET= install_h

.KEEP_STATE:

.PARALLEL:      $(PARALLEL_KMODS) $(XMODS) config $(LINT_DEPS)
.PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(SVVS) $(XMODS) \
                $(CLOSED_XMODS) config $(LINT_DEPS)

def all install clean clobber modlist: $(KMODS) $(XMODS) config
def all install clean clobber modlist: $(KMODS) $(CLOSED_KMODS) $(SVVS) \
        $(XMODS) $(CLOSED_XMODS) config

modlintlib clean.lint: $(LINT_KMODS) $(XMODS)
modlintlib clean.lint: $(LINT_KMODS) $(CLOSED_LINT_KMODS) $(SVVS) \
        $(XMODS) $(CLOSED_XMODS)

$(KMODS) config:        FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

$(CLOSED_KMODS):        FRC
        cd $(CLOSED)/uts/sparc/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

$(XMODS):       FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

$(SVVS) $(CLOSED_XMODS):        FRC
        @if [ -f $(CLOSED)/uts/sparc/$@/Makefile  ]; then \
                cd $(CLOSED)/uts/sparc/$@; pwd; \
                    $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

install_h check:        FRC
        @cd asm; pwd; $(MAKE) $(TARGET)
        @cd sys; pwd; $(MAKE) $(TARGET)
        @cd v7/sys; pwd; $(MAKE) $(TARGET)
        @cd v9/sys; pwd; $(MAKE) $(TARGET)

#
#       Full kernel lint target.
#
LINT_TARGET     = globallint

globallint:
        @-$(ECHO) "\nFULL KERNEL: global crosschecks:"
        @-$(LINT) $(LINTFLAGS) $(LINT_LIBS) 2>&1 | $(LGREP.2)

lint:   modlintlib .WAIT $(LINT_DEPS)

include ../Makefile.targ
```

```
**********************************************************
   13057 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sparc/Makefile.sparc.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

# Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2013 Andrew Stormont.  All rights reserved.


#
#       This makefile contains the common definitions for all sparc
#       implementation architecture independent modules.
#


#
#       Define supported builds
#
DEF_BUILDS      = $(DEF_BUILDS64)
ALL_BUILDS      = $(ALL_BUILDS64)

#
#       Everybody needs to know how to build modstubs.o and to locate unix.o.
#       Note that unix.o must currently be selected from among the possible
#       "implementation architectures". Note further, that unix.o is only
#       used as an optional error check for undefines so (theoretically)
#       any "implementation architectures" could be used. We choose sun4u
#       because it is the reference port.
#
UNIX_DIR         = $(UTSBASE)/sun4u/unix
GENLIB_DIR       = $(UTSBASE)/sun4u/genunix
IPDRV_DIR        = $(UTSBASE)/sparc/ip
MODSTUBS_DIR     = $(UNIX_DIR)
DSF_DIR          = $(UNIX_DIR)
LINTS_DIR        = $(OBJS_DIR)
LINT_LIB_DIR     = $(UTSBASE)/sparc/lint-libs/$(OBJS_DIR)

UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
GENLIB           = $(UTSBASE)/sun4u/lint-libs/$(OBJS_DIR)/libgenunix.so

LINT_LIB_32      = $(UTSBASE)/sun4u/lint-libs/$(OBJS_DIR)/llib-lunix.ln
GEN_LINT_LIB_32  = $(UTSBASE)/sun4u/lint-libs/$(OBJS_DIR)/llib-lgenunix.ln
```



```
LINT_LIB_64      = $(UTSBASE)/sun4u/lint-libs/$(OBJS_DIR)/llib-lunix.ln
GEN_LINT_LIB_64  = $(UTSBASE)/sun4u/lint-libs/$(OBJS_DIR)/llib-lgenunix.ln

LINT_LIB         = $(LINT_LIB_$(CLASS))
GEN_LINT_LIB     = $(GEN_LINT_LIB_$(CLASS))

LINT32_DIRS      = $(LINT32_BUILDS:%=$(UTSBASE)/sparc/lint-libs/%)
LINT32_FILES     = $(LINT32_DIRS:%=%/llib-l$(MODULE).ln)

LINT64_DIRS      = $(LINT64_BUILDS:%=$(UTSBASE)/sparc/lint-libs/%)
LINT64_FILES     = $(LINT64_DIRS:%=%/llib-l$(MODULE).ln)

#
#       Include the makefiles which define build rule templates, the
#       collection of files per module, and a few specific flags. Note
#       that order is significant, just as with an include path. The
#       first build rule template which matches the files name will be
#       used. By including these in order from most machine dependent
#       to most machine independent, we allow a machine dependent file
#       to be used in preference over a machine independent version
#       (Such as a machine specific optimization, which preserves the
#       interfaces.)
#
include $(UTSBASE)/sparc/Makefile.files
include $(UTSBASE)/sparc/v9/Makefile.files
include $(UTSTREE)/sun/Makefile.files
include $(UTSTREE)/common/Makefile.files

#
# ----- TRANSITIONAL SECTION --------------------------------------------------
#

#
#       Not everything which *should* be a module is a module yet. The
#       following is a list of such objects which are currently part of
#       genunix but which might someday become kmods.   This must be
#       defined before we include Makefile.uts, or else genunix's build
#       won't be as parallel as we might like.
#
NOT_YET_KMODS    = $(OLDPTY_OBJS) $(PTY_OBJS) $(VCONS_CONF_OBJS) $(MOD_OBJS)

#
# ----- END OF TRANSITIONAL SECTION -------------------------------------------
#
#       Include machine independent rules. Note that this does not imply
#       that the resulting module from rules in Makefile.uts is machine
#       independent. Only that the build rules are machine independent.
#
include $(UTSBASE)/Makefile.uts

#
#       machine specific optimization, override default in Makefile.master
#
XARCH_32        = -xarch=v8
XARCH_64        = -m64
XARCH           = $(XARCH_$(CLASS))

COPTIMIZE_32    = -xO3
COPTIMIZE_64    = -xO3
COPTIMIZE       = $(COPTIMIZE_$(CLASS))

CCMODE          = -Xa

CFLAGS_32       = -xcg92
CFLAGS_64       = -xchip=ultra $(CCABS32) $(CCREGSYM)
```

```
125 CFLAGS          = $(CFLAGS_$(CLASS))

127 CFLAGS          += $(XARCH)
128 CFLAGS          += $(COPTIMIZE)
129 CFLAGS          += $(EXTRA_CFLAGS)
130 CFLAGS          += $(XAOPT)
131 CFLAGS          += $(INLINES) -D_ASM_INLINES
132 CFLAGS          += $(CCMODE)
133 CFLAGS          += $(SPACEFLAG)
134 CFLAGS          += $(CERRWARN)
135 CFLAGS          += $(CTF_FLAGS_$(CLASS))
136 CFLAGS          += $(C99MODE)
137 CFLAGS          += $(CCUNBOUND)
138 CFLAGS          += $(CCSTATICSYM)
139 CFLAGS          += $(CC32BITCALLERS)
140 CFLAGS          += $(CCNOAUTOINLINE)
141 CFLAGS          += $(IROPTFLAG)
142 CFLAGS          += $(CGLOBALSTATIC)
143 CFLAGS          += -xregs=no%float
144 CFLAGS          += -xstrconst
145 CFLAGS          += $(CSOURCEDEBUGFLAGS)
146 CFLAGS          += $(CUSERFLAGS)

148 ASFLAGS         += $(XARCH)

150 LINT_DEFS_32    =
151 LINT_DEFS_64    = -m64
152 LINT_DEFS       += $(LINT_DEFS_$(CLASS))

154 #
155 #       The following must be defined for all implementations:
156 #
157 #       MODSTUBS:       Module stubs source file.
158 #
159 MODSTUBS         = $(UTSBASE)/sparc/ml/modstubs.s

161 #
162 #       Define the actual specific platforms - obviously none.
163 #
164 MACHINE_DEFS     =

166 #
167 #       Debugging level
168 #
169 #       Special knowledge of which special debugging options effect which
170 #       file is used to optimize the build if these flags are changed.
171 #
172 #       XXX: The above could possibly be done for more flags and files, but
173 #            is left as an experiment to the interested reader. Be forewarned,
174 #            that excessive use could lead to maintenance difficulties.
175 #
176 DEBUG_DEFS_OBJ32        =
177 DEBUG_DEFS_DBG32        = -DDEBUG
178 DEBUG_DEFS_OBJ64        =
179 DEBUG_DEFS_DBG64        = -DDEBUG
180 DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

182 DEBUG_COND_OBJ32        = $(POUND_SIGN)
183 DEBUG_COND_DBG32        =
184 DEBUG_COND_OBJ64        = $(POUND_SIGN)
185 DEBUG_COND_DBG64        =
186 IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

188 $(IF_DEBUG_OBJ)syscall.o        :=      DEBUG_DEFS      += -DSYSCALLTRACE
189 $(IF_DEBUG_OBJ)clock.o          :=      DEBUG_DEFS      += -DKSLICE=1
```



```
191 # Comment these out if you don't want dispatcher lock statistics.

193 # $(IF_DEBUG_OBJ)disp_lock.o    := DEBUG_DEFS   += -DDISP_LOCK_STATS

195 #
196 #       Collect the preprocessor definitions to be associated with *all*
197 #       files.
198 #
199 ALL_DEFS         = $(MACHINE_DEFS) $(DEBUG_DEFS) $(OPTION_DEFS)
200 #
201 #
202 #       The kernels modules which are "implementation architecture"
203 #       specific for this machine are enumerated below. Note that most
204 #       of these modules must exist (in one form or another) for each
205 #       architecture.
206 #
207 #       Common Drivers (usually pseudo drivers) (/kernel/drv):
208 #
209 DRV_KMODS       += aggr arp audio bl blkdev bofi clone cn conskbd consms cpuid
210 DRV_KMODS       += crypto cryptoadm devinfo dump
211 DRV_KMODS       += dtrace fasttrap fbt lockstat profile sdt systrace dcpc
212 DRV_KMODS       += fssnap icmp icmp6 ip ip6 ipnet ipsecah
213 DRV_KMODS       += ipsecesp iptun iwscn keysock kmdb kstat ksyms llc1
214 DRV_KMODS       += lofi
215 DRV_KMODS       += log logindmux kssl mm nca physmem pm poll pool
216 DRV_KMODS       += pseudo ptc ptm pts ptsl ramdisk random rsm rts sad
217 DRV_KMODS       += simnet softmac sppp sppptun sy sysevent sysmsg
218 DRV_KMODS       += spdsock
219 DRV_KMODS       += tcp tcp6 tl tnf ttymux udp udp6 wc winlock zcons
220 DRV_KMODS       += ippctl
221 DRV_KMODS       += dld
222 DRV_KMODS       += ipf
223 DRV_KMODS       += rpcib
224 DRV_KMODS       += dlpistub
225 DRV_KMODS       += vnic
226 DRV_KMODS       += xge
227 DRV_KMODS       += rds
228 DRV_KMODS       += rdsv3
229 DRV_KMODS       += chxge
230 DRV_KMODS       += smbsrv
231 DRV_KMODS       += vscan
232 DRV_KMODS       += nsmb
233 DRV_KMODS       += fm
234 DRV_KMODS       += nulldriver
235 DRV_KMODS       += bridge trill
236 DRV_KMODS       += bpf
237 DRV_KMODS       += dca

239 $(CLOSED_BUILD)CLOSED_DRV_KMODS += glm
240 $(CLOSED_BUILD)CLOSED_DRV_KMODS += isp
241 $(CLOSED_BUILD)CLOSED_DRV_KMODS += mpt
242 $(CLOSED_BUILD)CLOSED_DRV_KMODS += qus
243 $(CLOSED_BUILD)CLOSED_DRV_KMODS += se

239 #
240 #       Hardware Drivers in common space
241 #

243 DRV_KMODS       += afe
244 DRV_KMODS       += audio1575
245 DRV_KMODS       += audioens
246 DRV_KMODS       += audiols
247 DRV_KMODS       += audiop16x
248 DRV_KMODS       += audiopci
249 DRV_KMODS       += audiots
250 DRV_KMODS       += e1000g
```

```
251 DRV_KMODS       += efe
252 DRV_KMODS       += hxge
253 DRV_KMODS       += mxfe
254 DRV_KMODS       += rge
255 DRV_KMODS       += rtls
256 DRV_KMODS       += sfe
257 DRV_KMODS       += aac
258 DRV_KMODS       += igb
259 DRV_KMODS       += ixgbe
260 DRV_KMODS       += vr
261 DRV_KMODS       += mr_sas
268 $(CLOSED_BUILD)CLOSED_DRV_KMODS += ixgb
262 DRV_KMODS       += yge

264 #
265 #       Machine Specific Driver Modules (/kernel/drv):
266 #
267 DRV_KMODS       += audiocs
268 DRV_KMODS       += bge dmfe eri fas hme qfe
269 DRV_KMODS       += openeepr options sd ses st
270 DRV_KMODS       += ssd
271 DRV_KMODS       += ecpp
272 DRV_KMODS       += hid hubd ehci ohci uhci usb_mid usb_ia scsa2usb usbprn ugen
273 DRV_KMODS       += usbser usbsacm usbsksp usbsprl
274 DRV_KMODS       += usb_as usb_ac
275 DRV_KMODS       += usbskel
276 DRV_KMODS       += usbvc
277 DRV_KMODS       += usbftdi
278 DRV_KMODS       += wusb_df hwahc hwarc wusb_ca
279 DRV_KMODS       += usbecm
280 DRV_KMODS       += hci1394 av1394 scsa1394 dcam1394
281 DRV_KMODS       += sbp2
282 DRV_KMODS       += ib ibp eibnx eoib rdsib sdp iser daplt hermon tavor sol_ucma
283 DRV_KMODS       += sol_umad
284 DRV_KMODS       += pci_pci pcieb pcieb_bcm
285 DRV_KMODS       += i8042 kb8042 mouse8042
286 DRV_KMODS       += fcode
287 DRV_KMODS       += mpt_sas
288 DRV_KMODS       += socal
289 DRV_KMODS       += sgen
290 DRV_KMODS       += myri10ge
291 DRV_KMODS       += smp
292 DRV_KMODS       += dad
293 DRV_KMODS       += scsi_vhci
294 DRV_KMODS       += fcp
295 DRV_KMODS       += fcip
296 DRV_KMODS       += fcsm
297 DRV_KMODS       += fp
298 DRV_KMODS       += qlc
299 DRV_KMODS       += qlge
300 DRV_KMODS       += stmf
301 DRV_KMODS       += stmf_sbd
302 DRV_KMODS       += fct
303 DRV_KMODS       += fcoe
304 DRV_KMODS       += fcoet
305 DRV_KMODS       += fcoei
306 DRV_KMODS       += qlt
307 DRV_KMODS       += iscsit
308 DRV_KMODS       += pppt
309 DRV_KMODS       += ncall nsctl sdbc nskern sv
310 DRV_KMODS       += ii rdc rdcsrv rdcstub
311 DRV_KMODS       += iscsi
312 DRV_KMODS       += emlxs
313 DRV_KMODS       += oce
314 DRV_KMODS       += srpt
315 DRV_KMODS       += pmcs
```



```
316 DRV_KMODS       += pmcs8001fw

325 $(CLOSED_BUILD)CLOSED_DRV_KMODS += ifp
326 $(CLOSED_BUILD)CLOSED_DRV_KMODS += uata
327 $(CLOSED_BUILD)CLOSED_DRV_KMODS += usbser_edge

318 #
319 #       I/O framework test drivers
320 #
321 DRV_KMODS       += pshot
322 DRV_KMODS       += gen_drv
323 DRV_KMODS       += tvhci tphci tclient
324 DRV_KMODS       += emul64

326 #
327 # PCMCIA specific module(s)
328 #
329 DRV_KMODS       += pcs
330 MISC_KMODS      += busra cardbus dada pcmcia
331 DRV_KMODS       += pcic

333 # Add lvm
334 #
335 DRV_KMODS       += md
336 MISC_KMODS      += md_mirror md_stripe md_hotspares md_raid md_trans md_notify
337 MISC_KMODS      += md_sp

339 #
340 #       Exec Class Modules (/kernel/exec):
341 #
342 EXEC_KMODS      += aoutexec elfexec intpexec shbinexec javaexec

344 #
345 #       Scheduling Class Modules (/kernel/sched):
346 #
347 SCHED_KMODS     += RT TS RT_DPTBL TS_DPTBL IA FSS FX FX_DPTBL SDC

349 #
350 #       File System Modules (/kernel/fs):
351 #
352 FS_KMODS        += dev devfs fdfs fifofs hsfs lofs namefs nfs pcfs tmpfs zfs
353 FS_KMODS        += zut specfs udfs ufs autofs cachefs procfs sockfs mntfs
354 FS_KMODS        += ctfs objfs sharefs dcfs smbfs

356 #
357 #       Streams Modules (/kernel/strmod):
358 #
359 STRMOD_KMODS    += bufmod connld dedump ldterm ms pckt pfmod
360 STRMOD_KMODS    += pipemod ptem redirmod rpcmod rlmod telmod timod
361 STRMOD_KMODS    += spppasyn spppcomp
362 STRMOD_KMODS    += tirdwr ttcompat
363 STRMOD_KMODS    += usbkbm usbms usbwcm usb_ah
364 STRMOD_KMODS    += drcompat
365 STRMOD_KMODS    += cryptmod
366 STRMOD_KMODS    += vuid3ps2

368 #
369 #       'System' Modules (/kernel/sys):
370 #
371 SYS_KMODS       += c2audit
372 SYS_KMODS       += exacctsys
373 SYS_KMODS       += inst_sync kaio msgsys semsys shmsys sysacct pipe
374 SYS_KMODS       += doorfs pset acctctl portfs

376 #
377 #       'User' Modules (/kernel/misc):
```

```
378 #
379 MISC_KMODS      += ac97
380 MISC_KMODS      += bignum
381 MISC_KMODS      += consconfig gld ipc nfs_dlboot nfssrv scsi
382 MISC_KMODS      += strplumb swapgeneric tlimod
383 MISC_KMODS      += rpcsec rpcsec_gss kgssapi kmech_dummy
384 MISC_KMODS      += kmech_krb5
385 MISC_KMODS      += fssnap_if
386 MISC_KMODS      += hidparser kbtrans usba usba10 usbs49_fw
387 MISC_KMODS      += s1394
388 MISC_KMODS      += hpcsvc pcihp
389 MISC_KMODS      += rsmops
390 MISC_KMODS      += kcf
391 MISC_KMODS      += ksocket
392 MISC_KMODS      += ibcm
393 MISC_KMODS      += ibdm
394 MISC_KMODS      += ibdma
395 MISC_KMODS      += ibmf
396 MISC_KMODS      += ibtl
397 MISC_KMODS      += sol_ofs
398 MISC_KMODS      += idm
399 MISC_KMODS      += idmap
400 MISC_KMODS      += hook
401 MISC_KMODS      += neti
402 MISC_KMODS      += ctf
403 MISC_KMODS      += mac dls
404 MISC_KMODS      += cmlb
405 MISC_KMODS      += tem
406 MISC_KMODS      += pcicfg fcodem fcpci
407 MISC_KMODS      += scsi_vhci_f_sym scsi_vhci_f_tpgs scsi_vhci_f_asym_sun
408 MISC_KMODS      += scsi_vhci_f_sym_hds
409 MISC_KMODS      += scsi_vhci_f_tape scsi_vhci_f_tpgs_tape
410 MISC_KMODS      += fctl
411 MISC_KMODS      += emlxs_fw
412 MISC_KMODS      += qlc_fw_2200
413 MISC_KMODS      += qlc_fw_2300
414 MISC_KMODS      += qlc_fw_2400
415 MISC_KMODS      += qlc_fw_2500
416 MISC_KMODS      += qlc_fw_6322
417 MISC_KMODS      += qlc_fw_8100
418 MISC_KMODS      += spuni
419 MISC_KMODS      += hwa1480_fw uwba
420 MISC_KMODS      += mii

422 MISC_KMODS      += klmmod klmops

435 $(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_asym_lsi
436 $(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_asym_emc
437 $(CLOSED_BUILD)CLOSED_MISC_KMODS        += scsi_vhci_f_sym_emc

424 #
425 #       Software Cryptographic Providers (/kernel/crypto):
426 #
427 CRYPTO_KMODS    += aes
428 CRYPTO_KMODS    += arcfour
429 CRYPTO_KMODS    += blowfish
430 CRYPTO_KMODS    += des
431 CRYPTO_KMODS    += md4
432 CRYPTO_KMODS    += md5
433 CRYPTO_KMODS    += ecc
434 CRYPTO_KMODS    += rsa
435 CRYPTO_KMODS    += sha1
436 CRYPTO_KMODS    += sha2
437 CRYPTO_KMODS    += swrand

439 #
```



```
440 # IP Policy Modules (/kernel/ipp):
441 #
442 IPP_KMODS       += dlcosmk
443 IPP_KMODS       += flowacct
444 IPP_KMODS       += ipgpc
445 IPP_KMODS       += dscpmk
446 IPP_KMODS       += tokenmt
447 IPP_KMODS       += tswtclmt

449 #
450 # 'Dacf' modules (/kernel/dacf)
451 DACF_KMODS      += consconfig_dacf

453 #
454 #       SVVS Testing Modules (/kernel/strmod):
455 #
456 #       These are streams and driver modules which are not to be
457 #       delivered with a released system. However, during development
458 #       it is convenient to build and install the SVVS kernel modules.
459 #
460 SVVS_KMODS      += lmodb lmode lmodr lmodt svvslo tidg tivc tmux

477 $(CLOSED_BUILD)SVVS             += svvs

462 #
463 #       Modules eXcluded from the product:
464 #
465 XMODS           +=
483 $(CLOSED_BUILD)CLOSED_XMODS =          \
484         sdpib           \
485         wsdrv

467 #
468 #       'Dacf' Modules (/kernel/dacf):
469 #
470 DACF_KMODS      += net_dacf

472 #
473 #       MAC-Type Plugin Modules (/kernel/mac)
474 #
475 MAC_KMODS       += mac_6to4
476 MAC_KMODS       += mac_ether
477 MAC_KMODS       += mac_ipv4
478 MAC_KMODS       += mac_ipv6
479 MAC_KMODS       += mac_wifi
480 MAC_KMODS       += mac_ib

482 #
483 # socketmod (kernel/socketmod)
484 #
485 SOCKET_KMODS    += sockpfp
486 SOCKET_KMODS    += socksctp
487 SOCKET_KMODS    += socksdp
488 SOCKET_KMODS    += sockrds
489 SOCKET_KMODS    += ksslf

491 #
492 #       kiconv modules (/kernel/kiconv):
493 #
494 KICONV_KMODS    += kiconv_emea kiconv_ja kiconv_ko kiconv_sc kiconv_tc
```

```
**********************************************************
    4517 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sparc/sd/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# uts/sparc/sd/Makefile
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
#
#       This makefile drives the production of the sd driver kernel module.
#
#       sparc architecture dependent
#

#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = sd
OBJECTS         = $(SD_OBJS:%=$(OBJS_DIR)/%)
LINTS           = $(SD_OBJS:%.o=$(LINTS_DIR)/%.ln)
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
CONF_SRCDIR     = $(UTSBASE)/sun/io/scsi/targets
WARLOCK_OUT     = $(SD_OBJS:%.o=%.ll)
WARLOCK_OK      = $(MODULE).ok
WLCMD_DIR       = $(UTSBASE)/common/io/warlock

#
#       Include common rules.
#
include $(UTSBASE)/sparc/Makefile.sparc

#
#       Define targets
#
ALL_TARGET      = $(BINARY) $(SRC_CONFILE)
```



```
LINT_TARGET     = $(MODULE).lint
INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

#
# lint pass one enforcement
#
CFLAGS += $(CCVERBOSE)

#
# Define dependencies on scsi and cmlb
#
LDFLAGS += -dy -N misc/scsi -N misc/cmlb

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
LINTTAGS        += -erroff=E_STATIC_UNUSED
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON

CERRWARN        += -_gcc=-Wno-unused-variable
CERRWARN        += -_gcc=-Wno-unused-function
CERRWARN        += -_gcc=-Wno-unused-label
CERRWARN        += -_gcc=-Wno-parentheses
CERRWARN        += -_gcc=-Wno-type-limits
CERRWARN        += -_gcc=-Wno-uninitialized

#
#       Default build targets.
#
.KEEP_STATE:

all:            $(ALL_DEPS)

def:            $(DEF_DEPS)

clean:          $(CLEAN_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

clobber:        $(CLOBBER_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

lint:           $(LINT_DEPS)

modlintlib:     $(MODLINTLIB_DEPS)

clean.lint:     $(CLEAN_LINT_DEPS)

install:        $(INSTALL_DEPS)

#
#       Include common targets.
#
include $(UTSBASE)/sparc/Makefile.targ


#
#       Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test
```

```
126 #
127 # Warlock targets
128 #
129 # Note that in warlock_with_{esp,isp} it is important to load sd.ll
130 # before {isp,esp}.ll; the reason is that both have _init/_info/_fini
131 # and warlock can only handle one extern function by a given name;
132 # any loaded after the first are ignored.

134 SCSI_FILES = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)
135 CMLB_FILES = $(CMLB_OBJS:%.o=-l ../cmlb/%.ll)

137 WARLOCK_TARGETS = warlock_alone warlock_with_esp warlock_with_fas
138 $(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_isp warlock_with_glm \
139         warlock_with_uata warlock_with_mpt

139 warlock: $(WARLOCK_TARGETS)

141 warlock_alone: $(WARLOCK_OK)

143 $(WARLOCK_OK): $(WLCMD_DIR)/sd.wlcmd $(WARLOCK_OUT) scsi_files \
144         warlock_ddi.files cmlb_files
145         $(WARLOCK) -c $(WLCMD_DIR)/sd.wlcmd $(WARLOCK_OUT) $(SCSI_FILES) \
146                 $(CMLB_FILES) \
147                 -l ../warlock/ddi_dki_impl.ll
148         $(TOUCH) $@

150 %.ll: $(UTSBASE)/common/io/scsi/targets/%.c
151         $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

153 warlock_with_esp: $(WLCMD_DIR)/sd_with_esp.wlcmd $(WARLOCK_OUT) scsi_files \
154         esp_files warlock_ddi.files cmlb_files
155         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_esp.wlcmd \
156                 $(WARLOCK_OUT) ../esp/esp $(SCSI_FILES) $(CMLB_FILES) \
157                 -l ../warlock/ddi_dki_impl.ll

159 warlock_with_fas: $(WLCMD_DIR)/sd_with_fas.wlcmd $(WARLOCK_OUT) scsi_files \
160         fas_files warlock_ddi.files cmlb_files
161         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_fas.wlcmd \
162                 $(WARLOCK_OUT) ../fas/fas \
163                 ../fas/fas_callbacks \
164                 $(SCSI_FILES) $(CMLB_FILES) \
165                 -l ../warlock/ddi_dki_impl.ll

169 warlock_with_isp: $(WLCMD_DIR)/sd_with_isp.wlcmd $(WARLOCK_OUT) scsi_files \
170         isp_files warlock_ddi.files cmlb_files
171         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_isp.wlcmd \
172                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/isp/isp $(SCSI_FILES) \
173                 $(CMLB_FILES) \
174                 -l ../warlock/ddi_dki_impl.ll

176 warlock_with_glm: $(WLCMD_DIR)/sd_with_glm.wlcmd $(WARLOCK_OUT) scsi_files \
177         glm_files warlock_ddi.files cmlb_files
178         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_glm.wlcmd \
179                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/glm/glm $(SCSI_FILES) \
180                 $(CMLB_FILES) \
181                 -l ../warlock/ddi_dki_impl.ll

183 warlock_with_uata: $(WLCMD_DIR)/sd_with_uata.wlcmd $(WARLOCK_OUT) scsi_files \
184         uata_files warlock_ddi.files cmlb_files
185         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_uata.wlcmd \
186                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/uata/*.ll $(SCSI_FILES) \
187                 $(CMLB_FILES) \
188                 -l ../warlock/ddi_dki_impl.ll

190 warlock_with_mpt: $(WLCMD_DIR)/sd_with_mpt.wlcmd $(WARLOCK_OUT) scsi_files \
```



```
191         mpt_files warlock_ddi.files cmlb_files
192         $(WARLOCK) -c $(WLCMD_DIR)/sd_with_mpt.wlcmd \
193                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/mpt/*.ll $(SCSI_FILES)  \
194                 $(CMLB_FILES) \
195                 -l ../warlock/ddi_dki_impl.ll

167 cmlb_files:
168         @cd ../cmlb; pwd; $(MAKE) warlock


171 scsi_files:
172         @cd ../scsi; pwd; $(MAKE) warlock

174 esp_files:
175         @cd ../esp; pwd; $(MAKE) warlock

177 fas_files:
178         @cd ../fas; pwd; $(MAKE) warlock

210 isp_files:
211         @cd $(CLOSED)/uts/sparc/isp; pwd; $(MAKE) warlock

213 glm_files:
214         @cd $(CLOSED)/uts/sparc/glm; pwd; $(MAKE) warlock

216 uata_files:
217         @cd $(CLOSED)/uts/sparc/uata; pwd; $(MAKE) warlock

219 mpt_files:
220         @cd $(CLOSED)/uts/sparc/mpt; pwd; $(MAKE) warlock

180 warlock_ddi.files:
181         @cd ../warlock; pwd; $(MAKE) warlock
```

```
**********************************************************
    3926 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sparc/ses/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# uts/sparc/ses/Makefile
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
#
# This makefile drives the production of the ses driver kernel module.
#
# sparc architecture dependent
#

#
# Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
# Define the module and object file sets.
#
MODULE          = ses
OBJECTS         = $(SES_OBJS:%=$(OBJS_DIR)/%)
LINTS           = $(SES_OBJS:%.o=$(LINTS_DIR)/%.ln)
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
CONF_SRCDIR     = $(UTSBASE)/sun/io/scsi/targets
WARLOCK_OUT     = $(SES_OBJS:%.o=%.ll)
WARLOCK_OK      = $(MODULE).ok
WLCMD_DIR       = $(UTSBASE)/common/io/warlock

#
# Include common rules.
#
include $(UTSBASE)/sparc/Makefile.sparc

#
# Define targets
#
ALL_TARGET      = $(BINARY) $(SRC_CONFILE)
```



```
LINT_TARGET     = $(MODULE).lint
INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)
CLEANFILES      += $(WARLOCK_TARGETS)

#
# Define dependency on scsi
#
LDFLAGS += -dy -N misc/scsi

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON

#
# Default build targets.
#
.KEEP_STATE:

def:            $(DEF_DEPS)

all:            $(ALL_DEPS)

clean:          $(CLEAN_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

clobber:        $(CLOBBER_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

modlintlib:     $(MODLINTLIB_DEPS) lint64

clean.lint:     $(CLEAN_LINT_DEPS)

install:        $(INSTALL_DEPS)

#
#       Include common targets.
#
include $(UTSBASE)/sparc/Makefile.targ


#
# Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test

#
# Warlock targets
#
# XXX FIX ME.  this can be improved quite a bit

SES_FILES       = $(MODULE).ll
SCSI_FILES      = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)

WARLOCK_TARGETS = $(WARLOCK_OK) warlock_with_esp.ok warlock_with_fas.ok
$(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_isp.ok warlock_with_glm.ok

warlock: $(WARLOCK_TARGETS)
```

```
$(WARLOCK_OK): $(WLCMD_DIR)/$(MODULE).wlcmd $(SES_FILES)
        @cd ../warlock; $(MAKE) warlock
        @cd ../scsi; $(MAKE) warlock
        $(WARLOCK) -c $(WLCMD_DIR)/$(MODULE).wlcmd $(SES_FILES) ../warlock/scsi.
                -l ../warlock/ddi_dki_impl.ll  \
                $(SCSI_FILES)
        @ $(TOUCH) $@

warlock_with_esp.ok: $(WLCMD_DIR)/ses_with_esp.wlcmd $(SES_FILES)
        @cd ../warlock; $(MAKE) warlock
        @cd ../scsi; $(MAKE) warlock
        @cd ../esp; $(MAKE) warlock;
        $(WARLOCK) -c $(WLCMD_DIR)/ses_with_esp.wlcmd $(SES_FILES) ../esp/esp  \
                ../warlock/scsi.ll \
                -l ../warlock/ddi_dki_impl.ll \
                $(SCSI_FILES)
        @ $(TOUCH) $@

warlock_with_fas.ok: $(WLCMD_DIR)/ses_with_fas.wlcmd $(SES_FILES)
        @cd ../warlock; $(MAKE) warlock
        @cd ../scsi; $(MAKE) warlock
        @cd ../fas; $(MAKE) warlock;
        $(WARLOCK) -c $(WLCMD_DIR)/ses_with_fas.wlcmd $(SES_FILES) \
                ../fas/fas ../fas/fas_callbacks \
                ../warlock/scsi.ll \
                -l ../warlock/ddi_dki_impl.ll \
                $(SCSI_FILES)
        @ $(TOUCH) $@

warlock_with_isp.ok: $(WLCMD_DIR)/ses_with_isp.wlcmd $(SES_FILES)
        @cd ../warlock; $(MAKE) warlock
        @cd ../scsi; $(MAKE) warlock
        @cd $(CLOSED)/uts/sparc/isp; $(MAKE) warlock;
        $(WARLOCK) -c $(WLCMD_DIR)/ses_with_isp.wlcmd $(SES_FILES) \
                $(CLOSED)/uts/sparc/isp/isp \
                ../warlock/scsi.ll \
                -l ../warlock/ddi_dki_impl.ll \
                $(SCSI_FILES)
        @ $(TOUCH) $@

warlock_with_glm.ok: $(WLCMD_DIR)/ses_with_glm.wlcmd $(SES_FILES)
        @cd ../warlock; $(MAKE) warlock
        @cd ../scsi; $(MAKE) warlock
        @cd $(CLOSED)/uts/sparc/glm; $(MAKE) warlock;
        $(WARLOCK) -c $(WLCMD_DIR)/ses_with_glm.wlcmd $(SES_FILES) \
                $(CLOSED)/uts/sparc/glm/glm \
                ../warlock/scsi.ll \
                -l ../warlock/ddi_dki_impl.ll \
                $(SCSI_FILES)
        @ $(TOUCH) $@

%.ll: $(UTSBASE)/common/io/scsi/targets/%.c
        $(WLCC) $(CPPFLAGS) -o $@ $<
```

```
**********************************************************
    4201 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sparc/st/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# uts/sparc/st/Makefile
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2011 Bayard G. Bell. All rights reserved.
#
#       This makefile drives the production of the st driver kernel module.
#
#       sparc architecture dependent
#

#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = st
OBJECTS         = $(ST_OBJS:%=$(OBJS_DIR)/%)
LINTS           = $(ST_OBJS:%.o=$(LINTS_DIR)/%.ln)
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)
CONF_SRCDIR     = $(UTSBASE)/sun/io/scsi/targets
WARLOCK_OUT     = $(ST_OBJS:%.o=%.ll)
WARLOCK_OK      = $(MODULE).ok
WLCMD_DIR       = $(UTSBASE)/common/io/warlock

#
#       Include common rules.
#
include $(UTSBASE)/sparc/Makefile.sparc

#
#       Define targets
#
ALL_TARGET      = $(BINARY) $(SRC_CONFILE)
LINT_TARGET     = $(MODULE).lint
```



```
INSTALL_TARGET  = $(BINARY) $(ROOTMODULE) $(ROOT_CONFFILE)

#
# lint pass one enforcement
#
CFLAGS += $(CCVERBOSE)

#
# Define dependency on scsi
#
LDFLAGS += -dy -N misc/scsi

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON

CERRWARN        += -_gcc=-Wno-parentheses
CERRWARN        += -_gcc=-Wno-switch
CERRWARN        += -_gcc=-Wno-uninitialized

#
#       Default build targets.
#
.KEEP_STATE:

all:            $(ALL_DEPS)

def:            $(DEF_DEPS)

clean:          $(CLEAN_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

clobber:        $(CLOBBER_DEPS)
                $(RM) $(WARLOCK_OUT) $(WARLOCK_OK)

lint:           $(LINT_DEPS)

modlintlib:     $(MODLINTLIB_DEPS)

clean.lint:     $(CLEAN_LINT_DEPS)

install:        $(INSTALL_DEPS)

#       Include common targets.
#
include $(UTSBASE)/sparc/Makefile.targ

#
#       Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test

#
# Warlock targets
#
# Note that in warlock_with_{esp,isp} it is important to load st.ll
# before {isp,esp}.ll; the reason is that both have _init/_info/_fini
# and warlock can only handle one extern function by a given name;
```

```
125 # any loaded after the first are ignored.

127 SCSI_FILES = $(SCSI_OBJS:%.o=-l ../scsi/%.ll)

129 WARLOCK_TARGETS =  warlock_alone warlock_with_esp warlock_with_fas
130 $(CLOSED_BUILD)WARLOCK_TARGETS += warlock_with_isp warlock_with_glm

131 warlock: $(WARLOCK_TARGETS)

133 warlock_alone: $(WARLOCK_OK)

135 warlock_with_esp: $(WLCMD_DIR)/st_with_esp.wlcmd $(WARLOCK_OUT) scsi_files \
136         esp_files warlock_ddi.files
137         $(WARLOCK) -c $(WLCMD_DIR)/st_with_esp.wlcmd \
138                 $(WARLOCK_OUT) ../esp/esp $(SCSI_FILES)  \
139                 -l ../warlock/ddi_dki_impl.ll

141 warlock_with_fas: $(WLCMD_DIR)/st_with_fas.wlcmd $(WARLOCK_OUT) scsi_files \
142         fas_files warlock_ddi.files
143         $(WARLOCK) -c $(WLCMD_DIR)/st_with_fas.wlcmd \
144                 $(WARLOCK_OUT) \
145                 ../fas/fas ../fas/fas_callbacks \
146                 $(SCSI_FILES)  \
147                 -l ../warlock/ddi_dki_impl.ll

150 warlock_with_isp: $(WLCMD_DIR)/st_with_isp.wlcmd $(WARLOCK_OUT) scsi_files \
151         isp_files warlock_ddi.files
152         $(WARLOCK) -c $(WLCMD_DIR)/st_with_isp.wlcmd \
153                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/isp/isp $(SCSI_FILES)  \
154                 -l ../warlock/ddi_dki_impl.ll

156 warlock_with_glm: $(WLCMD_DIR)/st_with_glm.wlcmd $(WARLOCK_OUT) scsi_files \
157         glm_files warlock_ddi.files
158         $(WARLOCK) -c $(WLCMD_DIR)/st_with_glm.wlcmd \
159                 $(WARLOCK_OUT) $(CLOSED)/uts/sparc/glm/glm $(SCSI_FILES) \
160                 -l ../warlock/ddi_dki_impl.ll

149 scsi_files:
150         @cd ../scsi; pwd; $(MAKE) warlock

152 esp_files:
153         @cd ../esp; pwd; $(MAKE) warlock

155 fas_files:
156         @cd ../fas; pwd; $(MAKE) warlock

171 isp_files:
172         @cd $(CLOSED)/uts/sparc/isp; pwd; $(MAKE) warlock

174 glm_files:
175         @cd $(CLOSED)/uts/sparc/glm; pwd; $(MAKE) warlock

158 st.ok: $(WLCMD_DIR)/st.wlcmd st.ll st_conf.ll  scsi_files warlock_ddi.files
159         $(WARLOCK) -c $(WLCMD_DIR)/st.wlcmd $(WARLOCK_OUT) $(SCSI_FILES) \
160         -l ../warlock/ddi_dki_impl.ll
161         $(TOUCH) $@

163 %.ll: $(UTSBASE)/common/io/scsi/targets/%.c
164         $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

166 warlock_ddi.files:
167         @cd ../warlock; pwd; $(MAKE) warlock

169 scsi.files:
170         @cd ../scsi; pwd; $(MAKE) warlock
```

```
**********************************************************
    3645 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sparc/warlock/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright (c) 1998, 2010, Oracle and/or its affiliates. All rights reserved.
#
#       sparc architecture dependent
#
# uts/sparc/warlock/Makefile
#
#       Path to the base of the uts directory tree (usually /usr/src/uts).
#
UTSBASE = ../..

#
#       Define the module and object file sets.
#
MODULE          = warlock
ROOTMODULE      = $(ROOT_DRV_DIR)/$(MODULE)

.KEEP_STATE:

CFLAGS += -I../../common/sys -I../../sun/sys/scsi -D_KERNEL
#
#       Defines for local commands.
#
WARLOCK         = warlock
WLCC            = wlcc
TOUCH           = touch
TEST            = test

include $(UTSBASE)/sparc/Makefile.sparc

#
#       lock_lint rules
#
all:    warlock warlock.1394 warlock.ecpp warlock.scsi \
        warlock.usb warlock.ib warlock.sata warlock.wc

warlock: $(MODULE).ok

warlock.ok: ddi_dki_impl.ll scsi.ll
```



```
        $(TOUCH) $@

%.ll: $(UTSBASE)/common/io/warlock/%.c
        $(WLCC) $(CPPFLAGS) -DDEBUG -o $@ $<

warlock.usb:
        @cd ../usba; $(MAKE) clean; $(MAKE) warlock
        @cd ../ohci; $(MAKE) clean; $(MAKE) warlock
        @cd ../uhci; $(MAKE) clean; $(MAKE) warlock
        @cd ../ehci; $(MAKE) clean; $(MAKE) warlock
        @cd ../hid; $(MAKE) clean; $(MAKE) warlock
        @cd ../scsa2usb; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_ac; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_as; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_ah; $(MAKE) clean; $(MAKE) warlock
        @cd ../ugen; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_mid; $(MAKE) clean; $(MAKE) warlock
        @cd ../usb_ia; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbprn; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbser; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsksp; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsprl; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbsacm; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbecm; $(MAKE) clean; $(MAKE) warlock
        @cd ../usbskel; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/usbser_edge; \
                $(MAKE) clean; $(MAKE) warlock

warlock.scsi:
        @cd ../dad; $(MAKE) clean; $(MAKE) warlock
        @cd ../dada; $(MAKE) clean; $(MAKE) warlock
        @cd ../esp; $(MAKE) clean; $(MAKE) warlock
        @cd ../fas; $(MAKE) clean; $(MAKE) warlock
        @cd ../sd; $(MAKE) clean; $(MAKE) warlock
        @cd ../ses; $(MAKE) clean; $(MAKE) warlock
        @cd ../st; $(MAKE) clean; $(MAKE) warlock
        @cd ../ssd; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/uata; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/glm; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/mpt; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/ifp; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/isp; $(MAKE) clean; $(MAKE) warlock

warlock.1394:
        @cd ../s1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../hci1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../scsa1394; $(MAKE) clean; $(MAKE) warlock
        @cd ../av1394; $(MAKE) clean; $(MAKE) warlock

warlock.ecpp:
        @cd ../ecpp; $(MAKE) clean; $(MAKE) warlock

warlock.ib:
        @cd ../ibmf; $(MAKE) clean; $(MAKE) warlock
        @cd ../ib; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibtl; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibcm; $(MAKE) clean; $(MAKE) warlock
        @cd ../ibd; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/tavor; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/hermon; $(MAKE) clean; $(MAKE) warlock
$(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/daplt; $(MAKE) clean; $(MAKE) warlock

warlock.sata:
        @cd ../sata; $(MAKE) clean; $(MAKE) warlock
        @cd ../si3124; $(MAKE) clean; $(MAKE) warlock
        @cd ../ahci; $(MAKE) clean; $(MAKE) warlock
```

```
115         @cd ../nv_sata; $(MAKE) clean; $(MAKE) warlock
126 $(CLOSED_BUILD) @cd $(CLOSED)/uts/sparc/marvell88sx; \
127                 $(MAKE) clean; $(MAKE) warlock

117 warlock.wc:
118         @cd ../wc; $(MAKE) clean; $(MAKE) warlock
```

```
**********************************************************
    2989 Wed Oct 16 17:41:11 2013
new/usr/src/uts/sun/sys/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# include global definitions
include ../../../Makefile.master

HDRS=  \
avintr.h \
bootconf.h \
dkmpio.h                \
fdreg.h                 fdvar.h \
obpdefs.h               \
promif.h                promimpl.h \
ser_async.h             ser_zscc.h \
socalio.h               socalreg.h \
socal_cq_defs.h         socalmap.h              socalvar.h \
ttymux.h                ttymuxuser.h \
zsdev.h

SCSIADHDRS= \
fascmd.h        fasdma.h \
fasreg.h        fasvar.h        ifpio.h         sfvar.h

CLOSED_SCSIAD = $(CLOSED)/uts/sun/sys/scsi/adapters

CLOSED_SCSIADHDRS= \
ifpcmd.h        ifpmail.h       ifpreg.h        ifpvar.h \
ispcmd.h        ispmail.h       ispreg.h        ispvar.h

SCSITARGHDRS= \

ROOTDIR=        $(ROOT)/usr/include/sys
ROOTDIRS=       $(ROOTDIR) \
                $(ROOTDIR)/audio \
                $(ROOTDIR)/scsi/adapters \
                $(ROOTDIR)/scsi/targets
```



```
ROOTHDRS= $(HDRS:%=$(ROOTDIR)/%)
$(CLOSED_BUILD)ROOTHDRS += $(CLOSED_HDRS:%=$(ROOTDIR)/%)
ROOTFCHDRS= $(FCHDRS:%=$(ROOTDIR)/fc4/%)
ROOTSCSIADHDRS= $(SCSIADHDRS:%=$(ROOTDIR)/scsi/adapters/%)
$(CLOSED_BUILD)ROOTSCSIADHDRS += \
        $(CLOSED_SCSIADHDRS:%=$(ROOTDIR)/scsi/adapters/%)
ROOTSCSITARGHDRS= $(SCSITARGHDRS:%=$(ROOTDIR)/scsi/targets/%)


# install rules
$(ROOTDIR)/%: %
        $(INS.file)

$(ROOTDIR)/%: $(CLOSED)/uts/sun/sys/%
        $(INS.file)

$(ROOTDIR)/audio/%: audio/%
        $(INS.file)

$(ROOTDIR)/scsi/adapters/%: scsi/adapters/%
        $(INS.file)

$(ROOTDIR)/scsi/adapters/%: $(CLOSED_SCSIAD)/%
        $(INS.file)

$(ROOTDIR)/scsi/targets/%: scsi/targets/%
        $(INS.file)

# check files really don't exist

audio/%.check:          audio/%.h
        $(DOT_H_CHECK)

scsi/adapters/%.check:  scsi/adapters/%.h
        $(DOT_H_CHECK)

scsi/targets/%.check:   scsi/targets/%.h
        $(DOT_H_CHECK)

CHECKHDRS= \
        $(HDRS:%.h=%.check) \
        $(FCHDRS:%.h=fc4/%.check) \
        $(SCSIADHDRS:%.h=scsi/adapters/%.check) \
        $(SCSITARGHDRS:%.h=scsi/targets/%.check)

$(CLOSED_BUILD)CHECKHDRS += \
        $(CLOSED_HDRS:%.h=$(CLOSED)/uts/sun/sys/%.check) \
        $(CLOSED_SCSIADHDRS:%.h=$(CLOSED_SCSIAD)/%.check)

# headers which won't quite meet the standards...
#
# devops.h has a macro where the formal parameters to the macro are greater
# than 80 characters.  cpp (or the equivalent built into acomp) does not allow
# continuation line breaks in the formal parameter list.  This could be fixed
# by giving shorter names to the formal parameters, but the right fix is to
# fix cpp.  (Also, /* CSTYLED */ doesn't seem to fix this.
#
devops.check    := CSTYLE_TAIL = | grep -v "line > 80 characters" | true

.KEEP_STATE:

.PARALLEL:      $(CHECKHDRS) $(ROOTHDRS) $(ROOTAUDHDRS) $(ROOTAUDIMPLHDRS) \
                $(ROOTSCSIADHDRS) $(ROOTSCSITARGHDRS) $(ROOTXHDRS) \
                $(ROOTFCHDRS)
```

```
106 install_h:      $(ROOTDIRS) .WAIT $(ROOTHDRS) \
107                 $(ROOTAUDHDRS) $(ROOTAUDIMPLHDRS) \
108                 $(ROOTSCSIADHDRS) $(ROOTSCSITARGHDRS) $(ROOTFCHDRS)

110 $(ROOTDIRS):
111         $(INS.dir)

113 check:  $(CHECKHDRS)
```

```
**********************************************************
    7899 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#       This makefile drives the production of all implementation architecture
#       dependent modules for the sun4u architecture.
#

UTSBASE = ..

include Makefile.sun4u

#
#       The following are SPARC specific (rather than sun4u) specific modules
#       which are required for the sun4u kernel to completely lint. They are
#       not involved in the build in any other way. In order to minimize
#       build time, it is assumed that they are up to date. But since sun4u
#       is really a separate architecture we cannot use the v7 sparc modules.
#
SPARC_LIB_DIR    = $(UTSBASE)/sparc/lint-libs/$(OBJS_DIR)

SPARC_LINTS      =


#
#
#
LINT_LIBS        = $(LINT_LIB) \
                   $(LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(CLOSED_LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(SPARC_LINTS:%=$(SPARC_LIB_DIR)/llib-l%.ln)


def             :=      TARGET= def
def.prereq      :=      TARGET= def
all             :=      TARGET= all
all.prereq      :=      TARGET= all
install         :=      TARGET= install
install.prereq  :=      TARGET= all
```



```
install_h       :=      TARGET= install_h
install_h.prere :=      TARGET= install_h
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint
lint.prereq     :=      TARGET= lint
lintlib         :=      TARGET= lintlib
modlintlib      :=      TARGET= modlintlib
modlist         :=      TARGET= modlist
modlist modlist.sparc :=        NO_STATE= -K $$MODSTATE$$$$
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check

.KEEP_STATE:

.PARALLEL:      $(PARALLEL_KMODS) $(XMODS) \
                $(IMPLEMENTATIONS) \
.PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
                $(IMPLEMENTATIONS) $(CLOSED_IMPLEMENTATIONS) \
                modlist modlist.sparc

# Override for CPU_KMODS... they cannot be built
# in parallel
.NO_PARALLEL:   $(CPU_KMODS)

def all clean clobber clean.lint: genassym unix .WAIT \
        $(KMODS) $(XMODS) $(IMPLEMENTATIONS)
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
        $(IMPLEMENTATIONS) $(CLOSED_IMPLEMENTATIONS)

clobber: clobber.targ

# list the modules under sun4u.
modlist: unix $(KMODS) $(XMODS) $(IMPLEMENTATIONS:.WAIT=)
modlist: unix $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
        $(IMPLEMENTATIONS:.WAIT=) $(CLOSED_IMPLEMENTATIONS)

# list the modules for Install -k sun4u.
modlist.karch: modlist modlist.sparc

modlist.sparc:
        @cd $(SRC)/uts/sparc; pwd; $(MAKE) $(NO_STATE) modlist

install: install_platforms genassym unix .WAIT $(KMODS) \
        $(XMODS) $(IMPLEMENTATIONS)
install: install_platforms genassym unix .WAIT $(KMODS) $(CLOSED_KMODS) \
        $(XMODS) $(CLOSED_XMODS) $(IMPLEMENTATIONS) $(CLOSED_IMPLEMENTATIONS)

lintlib:        unix

modlintlib:     $(LINT_KMODS)
modlintlib:     $(LINT_KMODS) $(CLOSED_LINT_KMODS)

genassym unix $(KMODS): FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

#
# Privilege constants
#
# NOTE: The rules for generating priv_const.c file are shared between all
# processor architectures and should be kept in sync. If they are changed in
# this file make sure that x86 rules are updated as well.
#
PRIVS_C = $(UTSBASE)/common/os/priv_const.c

$(PRIVS_C): $(PRIVS_AWK) $(PRIVS_DEF)
```

```
115         $(NAWK) -f $(PRIVS_AWK) < $(PRIVS_DEF) cfile=$@

117 CLOBBERFILES += $(PRIVS_C)

119 #
120 # Prerequisites
121 #
122 # The uts/Makefile defines build parallelism for sun4 platforms such that sparc,
123 # sun4u and sun4v are all built in parallel. Also this Makefile specifies that
124 # all IMPLEMENTATIONS sun4u sub-platforms are built in parallel. This requires
125 # building certain parts before the parallel build can start. The uts/Makefile
126 # appends the '.prereq' string to the original target and executes this Makefile
127 # to build any prerequisites needed before the full parallel build can start.
128 # After that make continues with normal targets.
129 #
130 # Any build prerequisites for sun4 and IMPLEMENTATIONS builds should be
131 # described here.
132 #
133 # genassym is used to build dtrace and genunix, so it should be built first.
134 #
135 # priv_const.c is required to build genunix.
136 #
137 # genunix is used by everyone to ctfmerge with. Genunix is merged with sparc/ip
138 #   so as a side effect this dependency builds sparc/ip as part of the
139 #   prerequisites.
140 #
141 # unix is not required by itself but several sun4u platforms require
142 #   sun4u/platmod to be present. The easiest way to achieve this is to build
143 #   sun4u/unix first since sun4u/unix Makefile builds sun4u/platform correctly.
144 #   This causes full sun4u/unix to be built before all sun4u platforms and
145 #   before uts/sun4v and uts/sparc, but it acceptable since it is not spending
146 #   too much time building sun4u/unix.
147 #
148 all.prereq def.prereq install.prereq: genassym genunix unix

150 #
151 # Various sun4u platforms expect proto/root_sparc/platform/sun4u/include to be
152 # present. This is handled by running make install_h in sun4u/unix directory
153 # first.
154 #
155 install_h.prereq: FRC
156         @cd sys; pwd; $(MAKE) $(TARGET)

158 #
159 # sun4u/unix and sun4u/genunix should be linted first since sparc does global
160 # cross-check with these lint libraries. The sun4u/unix and sun4u/genunix can be
161 # linted in parallel.
162 #
163 LINT_PREREQ = unix.lint genunix.lint
164 lint.prereq: $(LINT_PREREQ)

166 .PARALLEL:  $(LINT_PREREQ)

168 $(LINT_PREREQ):
169         @cd $(@:%.lint=%); pwd; $(MAKE) $(TARGET)

171 #
172 # Nothing to do with any other prerequisites
173 #
174 %.prereq:

176 #
177 # Platform inter-dependencies
178 #
179 lw8: serengeti
```



```
181 quasar: darwin

183 #
184 # The genunix requires priv_const.c file to be generated first.
185 #
186 genunix: $(PRIVS_C)

188 #
189 # Rules
190 #

192 $(IMPLEMENTATIONS):     FRC
193         @cd $@; pwd; THISIMPL=$@ $(MAKE) $(NO_STATE) $(TARGET)

198 $(CLOSED_IMPLEMENTATIONS):      FRC
199         cd $(CLOSED)/uts/sun4u/$@; pwd; \
200             THISIMPL=$@ $(MAKE) $(NO_STATE) $(TARGET); \

195 $(XMODS):       FRC
196         @if [ -f $@/Makefile  ]; then \
197                 cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
198         else \
199                 true; \
200         fi

209 $(CLOSED_XMODS):        FRC
210         @if [ -f $(CLOSED)/uts/sun4u/$@/Makefile ]; then \
211                 cd $(CLOSED)/uts/sun4u/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
212         else \
213                 true; \
214         fi

216 $(CLOSED_KMODS):        FRC
217         cd $(CLOSED)/uts/sun4u/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

202 install_h check: install_platforms $(IMPLEMENTATIONS) \
220         $(CLOSED_IMPLEMENTATIONS) FRC
203         @cd sys; pwd; $(MAKE) $(TARGET)
204         @cd vm; pwd; $(MAKE) $(TARGET)

206 #
207 # Rules for the /platforms directories.  This is hardwired here because
208 # the first stage of the project (KBI) only implements the userland
209 # changes, but the only reasonable place to record the aliases is
210 # here in kernel land.
211 #
212 $(ROOT_PLAT_DIRS): $(ROOT_PLAT_DIR)
213         -$(INS.dir)

215 #
216 # create directories in /usr/platform/ for the implementations that are
217 # defined in $(IMPLEMENTED_PLATFORM)
218 #       (eg. SUNW,Ultra-1)
219 #
220 # Foreach $(IMPLEMENTED_PLATFORM) there can be a list of $(LINKED_PLATFORMS)
221 # that are linked to it.
222 #
223 $(USR_PLAT_DIR)/$(IMPLEMENTED_PLATFORM):        $(USR_PLAT_DIR)
224         -$(INS.dir)

226 #
227 # create the links in /usr/platform/ foreach $(LINKED_PLATFORMS)
228 # to it's corresponding $(IMPLEMENTED_PLATFORM).
229 #
230 PLATFORMS       = $(LINKED_PLATFORMS)
```

```
$(USR_PLAT_DIRS): $(USR_PLAT_DIR)
        $(INS.slink3)

PLATFORMS       += $(IMPLEMENTED_PLATFORM)

#
# Make the /platforms directories.  This is hardwired here because
# the first stage of the project (KBI) only implements the userland
# changes, but the only reasonable place to record the aliases is
# here in kernel land.
#
install_platforms:      $(ROOT_PSM_DIR) $(USR_PSM_DIR) \
                        $(ROOT_PLAT_DIRS) $(USR_PLAT_DIRS) \
                        $(USR_DESKTOP_DIR) $(USR_DESKTOP_INC_DIR) \
                        $(USR_DESKTOP_SBIN_DIR) $(USR_DESKTOP_LIB_DIR)

#
# rules for making include, sbin, lib dirs/links in
# /usr/platform/$(PLATFORM)/ for desktop platforms
#
$(USR_DESKTOP_INC_DIR):         $(USR_DESKTOP_DIR)
        $(INS.slink4)

$(USR_DESKTOP_SBIN_DIR):        $(USR_DESKTOP_DIR)
        $(INS.slink5)

$(USR_DESKTOP_LIB_DIR):         $(USR_DESKTOP_DIR)
        -$(INS.dir)

#
#       Full kernel lint target.
#
LINT_TARGET     = globallint

globallint:
        @pwd
        @-$(ECHO) "\nSUN4U KERNEL: global crosschecks:"
        @-$(LINT) $(LINTFLAGS) $(LINT_LIBS) 2>&1 | $(LGREP.2)

lint:   lintlib .WAIT modlintlib .WAIT $(SPARC_LINTS) $(LINT_DEPS) \
        $(IMPLEMENTATIONS) $(CPU_KMODS)
        $(IMPLEMENTATIONS) $(CLOSED_IMPLEMENTATIONS) $(CPU_KMODS)

include ../Makefile.targ

#
# Cross-reference customization: build a cross-reference over all of the
# sun4u-related directories.
#
XRDIRS  = ../sun4u ../sun4 ../sfmmu ../sparc ../sun ../common
SHARED_XRDIRS   = ../sun4u ../sun4 ../sfmmu ../sparc ../sun ../common
CLOSED_XRDIRS   = $(SHARED_XRDIRS:../%=../% ../../../closed/uts/%)
XRDIRS          = $(SHARED_XRDIRS)
$(CLOSED_BUILD)XRDIRS   = $(CLOSED_XRDIRS:../../../closed/uts/sfmmu=)

XRPRUNE = i86pc

cscope.out tags: FRC
        $(XREF) -x $@
```

```
**********************************************************
   13404 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/Makefile.sun4u.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2013 Andrew Stormont.  All rights reserved.
#
#       This makefile contains the common definitions for the sun4u unix
#       and all sun4u implementation architecture dependent modules.
#


#
#       Machine type (implementation architecture):
#
PLATFORM         = sun4u
PROMIF           = ieee1275
PSMBASE          = $(UTSBASE)/../psm

#
#       uname -m value
#
UNAME_M         = $(PLATFORM)

#
# Definitions for the platform-specific /platform directories.
#
# PLATFORMS designates those sun4u machines which have no platform
# specific code.
#
# IMPLEMENTATIONS is used to designate sun4u machines which do have
# platform specific modules (perhaps including their own unix).  All
# code specific to a given implementation resides in the appropriately
# named subdirectory.   This requires these platforms to have their
# own Makefiles to define ROOT_PLAT_DIRS, USR_PLAT_DIRS, etc.
#
# So if we had an implementation named 'foo', we would need the following
# Makefiles in the foo subdirectory:
#
#       sun4u/foo/Makefile
```



```
#       sun4u/foo/Makefile.foo
#       sun4u/foo/Makefile.targ
#

#
# /usr/platform/$(IMPLEMENTED_PLATFORM) is created as a directory that
# all the $(LINKED_PLATFORMS) link to.
#
IMPLEMENTED_PLATFORM    = SUNW,Ultra-2

LINKED_PLATFORMS        += SUNW,Ultra-30
LINKED_PLATFORMS        += SUNW,Ultra-60

#
# all PLATFORMS that do not belong in the $(IMPLEMENTATIONS) list
# ie. all desktop platforms
#
PLATFORMS                = $(IMPLEMENTED_PLATFORM)
PLATFORMS               += $(LINKED_PLATFORMS)

ROOT_PLAT_DIRS          = $(PLATFORMS:%=$(ROOT_PLAT_DIR)/%)
USR_PLAT_DIRS           = $(PLATFORMS:%=$(USR_PLAT_DIR)/%)

USR_DESKTOP_DIR         = $(USR_PLAT_DIR)/$(IMPLEMENTED_PLATFORM)
USR_DESKTOP_INC_DIR     = $(USR_DESKTOP_DIR)/include
USR_DESKTOP_SBIN_DIR    = $(USR_DESKTOP_DIR)/sbin
USR_DESKTOP_LIB_DIR     = $(USR_DESKTOP_DIR)/lib

#
# Welcome to SPARC V9.
#

#
#       Define supported builds
#
DEF_BUILDS      = $(DEF_BUILDS64)
ALL_BUILDS      = $(ALL_BUILDS64)

#
#       Everybody needs to know how to build modstubs.o and to locate unix.o
#
UNIX_DIR         = $(UTSBASE)/$(PLATFORM)/unix
GENLIB_DIR       = $(UTSBASE)/$(PLATFORM)/genunix
MODSTUBS_DIR     = $(UNIX_DIR)
DSF_DIR          = $(UTSBASE)/$(PLATFORM)/genassym
LINTS_DIR        = $(OBJS_DIR)
LINT_LIB_DIR     = $(UTSBASE)/$(PLATFORM)/lint-libs/$(OBJS_DIR)

DTRACESTUBS_O    = $(OBJS_DIR)/dtracestubs.o
DTRACESTUBS      = $(OBJS_DIR)/libdtracestubs.so

UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
GENLIB           = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so

LINT_LIB         = $(LINT_LIB_DIR)/llib-lunix.ln
GEN_LINT_LIB     = $(LINT_LIB_DIR)/llib-lgenunix.ln

LINT64_DIRS      = $(LINT64_BUILDS:%=$(UTSBASE)/$(PLATFORM)/lint-libs/%)
LINT64_FILES     = $(LINT64_DIRS:%=%/llib-l$(MODULE).ln)

#
#       cpu and platform modules need to know how to build their own symcheck mo
#
PLATMOD          = platmod
PLATLIB          = $(PLAT_DIR)/$(OBJS_DIR)/libplatmod.so
```

```
126 CPUNAME          = cpu
127 CPULIB           = $(CPU_DIR)/$(OBJS_DIR)/libcpu.so

129 SYM_MOD          = $(OBJS_DIR)/unix.sym

131 #
132 #       Include the makefiles which define build rule templates, the
133 #       collection of files per module, and a few specific flags. Note
134 #       that order is significant, just as with an include path. The
135 #       first build rule template which matches the files name will be
136 #       used. By including these in order from most machine dependent
137 #       to most machine independent, we allow a machine dependent file
138 #       to be used in preference over a machine independent version
139 #       (Such as a machine specific optimization, which preserves the
140 #       interfaces.)
141 #
142 include $(UTSBASE)/sun4/Makefile.files
143 include $(UTSTREE)/$(PLATFORM)/Makefile.files
144 include $(UTSBASE)/sfmmu/Makefile.files
145 include $(UTSBASE)/sparc/v9/Makefile.files
146 include $(UTSBASE)/sparc/Makefile.files
147 include $(UTSTREE)/sun/Makefile.files
148 include $(SRC)/psm/promif/$(PROMIF)/common/Makefile.files
149 include $(SRC)/psm/promif/$(PROMIF)/$(PLATFORM)/Makefile.files
150 include $(UTSTREE)/common/Makefile.files

152 #
153 #       Include machine independent rules. Note that this does not imply
154 #       that the resulting module from rules in Makefile.uts is machine
155 #       independent. Only that the build rules are machine independent.
156 #
157 include $(UTSBASE)/Makefile.uts

159 # These come after Makefile.uts.
159 # These come after Makefile.uts (for CLOSED_BUILD).
160 IMPLEMENTATIONS          = tazmo
161 IMPLEMENTATIONS         += starfire
162 IMPLEMENTATIONS         += javelin
163 IMPLEMENTATIONS         += darwin
164 IMPLEMENTATIONS         += quasar
165 IMPLEMENTATIONS         += grover
166 IMPLEMENTATIONS         += enchilada
167 IMPLEMENTATIONS         += taco
168 IMPLEMENTATIONS         += mpxu
169 IMPLEMENTATIONS         += excalibur
170 IMPLEMENTATIONS         += montecarlo
171 IMPLEMENTATIONS         += serengeti
172 IMPLEMENTATIONS         += littleneck
173 IMPLEMENTATIONS         += starcat
174 IMPLEMENTATIONS         += daktari
175 IMPLEMENTATIONS         += cherrystone
176 IMPLEMENTATIONS         += fjlite
177 IMPLEMENTATIONS         += snowbird
178 IMPLEMENTATIONS         += schumacher
179 IMPLEMENTATIONS         += blade
180 IMPLEMENTATIONS         += boston
181 IMPLEMENTATIONS         += seattle
182 IMPLEMENTATIONS         += chicago
183 IMPLEMENTATIONS         += sunfire
184 IMPLEMENTATIONS         += lw8
185 IMPLEMENTATIONS         += makaha
186 IMPLEMENTATIONS         += opl
187 IMPLEMENTATIONS         += lw2plus

189 $(CLOSED_BUILD)CLOSED_IMPLEMENTATIONS   = chalupa
```



```
190 $(CLOSED_BUILD)CLOSED_IMPLEMENTATIONS   += ents

189 #
190 #       machine specific optimization, override default in Makefile.master
191 #
192 CC_XARCH        = -m64 -xarch=sparcvis
193 AS_XARCH        = -xarch=v9a
194 COPTIMIZE       = -xO3
195 CCMODE          = -Xa

197 CFLAGS          = -xchip=ultra $(CCABS32) $(CCREGSYM)
198 CFLAGS          += $(CC_XARCH)
199 CFLAGS          += $(COPTIMIZE)
200 CFLAGS          += $(EXTRA_CFLAGS)
201 CFLAGS          += $(XAOPT)
202 CFLAGS          += $(INLINES) -D_ASM_INLINES
203 CFLAGS          += $(CCMODE)
204 CFLAGS          += $(SPACEFLAG)
205 CFLAGS          += $(CERRWARN)
206 CFLAGS          += $(CTF_FLAGS_$(CLASS))
207 CFLAGS          += $(C99MODE)
208 CFLAGS          += $(CCUNBOUND)
209 CFLAGS          += $(CCNOAUTOINLINE)
210 CFLAGS          += $(CCSTATICSYM)
211 CFLAGS          += $(CC32BITCALLERS)
212 CFLAGS          += $(IROPTFLAG)
213 CFLAGS          += $(CGLOBALSTATIC)
214 CFLAGS          += -xregs=no%float
215 CFLAGS          += -xstrconst
216 CFLAGS          += $(CSOURCEDEBUGFLAGS)
217 CFLAGS          += $(CUSERFLAGS)

219 ASFLAGS         += $(AS_XARCH)

221 AS_INC_PATH     += -I$(DSF_DIR)/$(OBJS_DIR)

223 LINT_KMODS      += $(GENUNIX_KMODS)

225 LINT_DEFS = -m64

227 #
228 #       The following must be defined for all implementations:
229 #
230 #       MAPFILE:                ld mapfile for the build of kernel/unix.
231 #       MODSTUBS:               Module stubs source file.
232 #       GENCONST_SRC:           genconst.c
233 #       OFFSETS:                offsets.in
234 #       PLATFORM_OFFSETS:       Platform specific mach_offsets.in
235 #       FDOFFSETS:              fd_offsets.in
236 #
237 MAPFILE          = $(UTSBASE)/sun4/conf/Mapfile
238 MODSTUBS         = $(UTSBASE)/sparc/ml/modstubs.s
239 GENCONST_SRC     = $(UTSBASE)/sun4/ml/genconst.c
240 OFFSETS          = $(UTSBASE)/sun4/ml/offsets.in
241 PLATFORM_OFFSETS = $(UTSBASE)/sun4u/ml/mach_offsets.in
242 FDOFFSETS        = $(UTSBASE)/sun/io/fd_offsets.in

244 #
245 #       Define the actual specific platforms
246 #

248 MACHINE_DEFS    = -D$(PLATFORM) -D_MACHDEP -DSFMMU

250 #
251 #       Software workarounds for hardware "features"
252 #
```

```
include $(UTSBASE)/$(PLATFORM)/Makefile.workarounds

#
#       Debugging level
#
#       Special knowledge of which special debugging options effect which
#       file is used to optimize the build if these flags are changed.
#
#       XXX: The above could possibly be done for more flags and files, but
#            is left as an experiment to the interested reader. Be forewarned,
#            that excessive use could lead to maintenance difficulties.
#
#       Note: kslice can be enabled for the sun4u, but is disabled by default
#             in all cases.
#

DEBUG_DEFS_OBJ64        =
DEBUG_DEFS_DBG64        = -DDEBUG
DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

DEBUG_COND_OBJ64        = $(POUND_SIGN)
DEBUG_COND_DBG64        =
IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

$(IF_DEBUG_OBJ)trap.o           :=      DEBUG_DEFS      += -DTRAPDEBUG
$(IF_DEBUG_OBJ)mach_trap.o      :=      DEBUG_DEFS      += -DTRAPDEBUG
$(IF_DEBUG_OBJ)syscall_trap.o   :=      DEBUG_DEFS      += -DSYSCALLTRACE
$(IF_DEBUG_OBJ)clock.o          :=      DEBUG_DEFS      += -DKSLICE=0

IF_TRAPTRACE_OBJ = $(IF_DEBUG_OBJ)
# comment this out for a non-debug kernel with TRAPTRACE
#IF_TRAPTRACE_OBJ = $(OBJS_DIR)/

$(IF_TRAPTRACE_OBJ)mach_locore.o        :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mlsetup.o            :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)syscall_trap.o       :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)startup.o            :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mach_startup.o       :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mp_startup.o         :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)cpu_states.o         :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mach_cpu_states.o    :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)interrupt.o          :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mach_interrupt.o     :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)sfmmu_asm.o          :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)trap_table.o         :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)xc.o                 :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mach_xc.o            :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)wbuf.o               :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)trap.o               :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)mach_trap.o          :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)x_call.o             :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)spitfire_asm.o       :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)us3_common_asm.o     :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)us3_cheetah_asm.o    :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)us3_cheetahplus_asm.o :=     DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)us3_jalapeno_asm.o   :=      DEBUG_DEFS      += -DTRAPTRACE
$(IF_TRAPTRACE_OBJ)opl_olympus_asm.o    :=      DEBUG_DEFS      += -DTRAPTRACE

# Comment these out if you don't want dispatcher lock statistics.

#$(IF_DEBUG_OBJ)lock_prim.o     := DEBUG_DEFS   += -DDISP_LOCK_STATS
#$(IF_DEBUG_OBJ)disp.o          := DEBUG_DEFS   += -DDISP_LOCK_STATS

# Comment these out if you don't want dispatcher debugging
```



```
#$(IF_DEBUG_OBJ)lock_prim.o     := DEBUG_DEFS   += -DDISP_DEBUG

#
#       Collect the preprocessor definitions to be associated with *all*
#       files.
#
ALL_DEFS         = $(MACHINE_DEFS) $(WORKAROUND_DEFS) $(DEBUG_DEFS) \
                   $(OPTION_DEFS)
GENCONST_DEFS    = $(MACHINE_DEFS) $(OPTION_DEFS)

#
# ----- TRANSITIONAL SECTION --------------------------------------------------
#

#
#       Not everything which *should* be a module is a module yet. The
#       following is a list of such objects which are currently part of
#       the base kernel but should soon become kmods.
#
MACH_NOT_YET_KMODS       = $(AUTOCONF_OBJS)

#
# ----- END OF TRANSITIONAL SECTION -------------------------------------------
#

#
#       The kernels modules which are "implementation architecture"
#       specific for this machine are enumerated below. Note that most
#       of these modules must exist (in one form or another) for each
#       architecture.
#
#       Common Drivers (usually pseudo drivers) (/kernel/drv):
#

#
#       Machine Specific Driver Modules (/kernel/drv):
#
#       XXX: How many of these are really machine specific?
#
DRV_KMODS       += bbc_beep
DRV_KMODS       += cpc
DRV_KMODS       += fd
DRV_KMODS       += rootnex sbusmem upa64s zs zsh
DRV_KMODS       += sbus
DRV_KMODS       += pcisch pcipsy simba
DRV_KMODS       += px
DRV_KMODS       += ebus
DRV_KMODS       += su
DRV_KMODS       += tod
DRV_KMODS       += power
DRV_KMODS       += epic
DRV_KMODS       += grbeep
DRV_KMODS       += pcf8584 max1617 seeprom tda8444 pca9556
DRV_KMODS       += ics951601 adm1031
DRV_KMODS       += lm75 ltc1427 pcf8591 pcf8574 ssc050 ssc100
DRV_KMODS       += pic16f819
DRV_KMODS       += pic16f747
DRV_KMODS       += adm1026
DRV_KMODS       += us
DRV_KMODS       += ppm schppm jbusppm
DRV_KMODS       += mc-us3
DRV_KMODS       += mc-us3i
DRV_KMODS       += smbus
DRV_KMODS       += db21554
DRV_KMODS       += gpio_87317
DRV_KMODS       += isadma
```

```
385 DRV_KMODS       += sbbc
386 DRV_KMODS       += pmubus
387 DRV_KMODS       += pmugpio
388 DRV_KMODS       += pmc
389 DRV_KMODS       += trapstat
390 DRV_KMODS       += rmc_comm
391 DRV_KMODS       += rmcadm
392 DRV_KMODS       += rmclomv
393 DRV_KMODS       += sf
394 DRV_KMODS       += nxge
395 DRV_KMODS       += i2bsc
396 DRV_KMODS       += mem_cache

401 $(CLOSED_BUILD)CLOSED_DRV_KMODS += ctsmc
402 $(CLOSED_BUILD)CLOSED_DRV_KMODS += m1535ppm
403 $(CLOSED_BUILD)CLOSED_DRV_KMODS += memtest
404 $(CLOSED_BUILD)CLOSED_DRV_KMODS += mi2cv
405 $(CLOSED_BUILD)CLOSED_DRV_KMODS += smbus_ara

398 #
399 #       Exec Class Modules (/kernel/exec):
400 #
401 EXEC_KMODS      +=

403 #
404 #       Scheduling Class Modules (/kernel/sched):
405 #
406 SCHED_KMODS     +=

408 #
409 #       File System Modules (/kernel/fs):
410 #
411 FS_KMODS        +=

413 #
414 #       Streams Modules (/kernel/strmod):
415 #
416 STRMOD_KMODS    += kb

418 #
419 #       'System' Modules (/kernel/sys):
420 #
421 SYS_KMODS       +=

423 #
424 #       'User' Modules (/kernel/misc):
425 #
426 MISC_KMODS      += bignum
427 MISC_KMODS      += obpsym bootdev vis cpr platmod md5 sha1 i2c_svc
428 MISC_KMODS      += sbd

430 MISC_KMODS      += opl_cfg
431 MISC_KMODS      += zuluvm
432 MISC_KMODS      += gptwo_cpu gptwocfg
433 MISC_KMODS      += pcie

435 #
436 #       Brand modules
437 #
438 BRAND_KMODS     += sn1_brand s10_brand

440 #
441 #       Software Cryptographic Providers (/kernel/crypto):
442 #
443 CRYPTO_KMODS    += aes
444 CRYPTO_KMODS    += arcfour
```



```
445 CRYPTO_KMODS    += des

447 #
448 #       generic-unix module (/kernel/genunix):
449 #
450 GENUNIX_KMODS   += genunix

452 #       'User' "Modules" excluded from the Full Kernel lint target:
453 #
463 $(CLOSED_BUILD)CLOSED_NLMISC_KMODS      += forthdebug

455 #
456 #       Modules eXcluded from the product:
457 #
458 XMODS           +=

460 #
461 #       cpu modules
462 #
463 CPU_KMODS       += cheetah cheetahplus jalapeno serrano spitfire hummingbird

465 #
466 #       sun4u 'TOD' Modules (/platform/.../kernel/tod):
467 #
468 TOD_KMODS       += todds1287 todds1337 todmostek todstarfire
469 TOD_KMODS       += todm5819 todblade todbq4802 todsg todopl
470 TOD_KMODS       += todm5819p_rmc todstarcat

482 $(CLOSED_BUILD)CLOSED_TOD_KMODS += todm5823

472 #
473 #       Performance Counter BackEnd Modules (/usr/kernel/pcbe):
474 #
475 PCBE_KMODS      += us234_pcbe
476 PCBE_KMODS      += opl_pcbe
```

```
**********************************************************
    3824 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/blade/Makefile.blade.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
#ident  "%Z%%M% %I%     %E% SMI"
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#       Global definitions for sun4u jbos-blade implementation specific modules.
#


#
#       Define directories.
#
ROOT_BLADE_DIR          = $(ROOT_PLAT_DIR)/SUNW,Serverblade1
ROOT_BLADE_MOD_DIR      = $(ROOT_BLADE_DIR)/kernel
ROOT_BLADE_MISC_DIR_32  = $(ROOT_BLADE_DIR)/kernel/misc
ROOT_BLADE_MISC_DIR_64  = $(ROOT_BLADE_MISC_DIR_32)/$(SUBDIR64)
ROOT_BLADE_KERN_DIR_32 = $(ROOT_BLADE_MOD_DIR)
ROOT_BLADE_KERN_DIR_64 = $(ROOT_BLADE_MOD_DIR)/$(SUBDIR64)
ROOT_BLADE_DRV_DIR_32   = $(ROOT_BLADE_MOD_DIR)/drv
ROOT_BLADE_DRV_DIR_64   = $(ROOT_BLADE_MOD_DIR)/drv/$(SUBDIR64)

ROOT_BLADE_KERN_DIR     = $(ROOT_BLADE_KERN_DIR_$(CLASS))
ROOT_BLADE_DRV_DIR      = $(ROOT_BLADE_DRV_DIR_$(CLASS))
ROOT_BLADE_MISC_DIR     = $(ROOT_BLADE_MISC_DIR_$(CLASS))

ROOT_PLAT_MOD_DIRS      += $(ROOT_BLADE_MOD_DIR)
ROOT_PLAT_MISC_DIRS     += $(ROOT_BLADE_MISC_DIR)


USR_SUN4U_PLAT_DIR      = $(USR_PLAT_DIR)/sun4u
USR_BLADE_DIR           = $(USR_PLAT_DIR)/SUNW,Serverblade1
USR_BLADE_INC_DIR       = $(USR_BLADE_DIR)/include
USR_BLADE_ISYS_DIR      = $(USR_BLADE_INC_DIR)/sys
USR_BLADE_SBIN_DIR      = $(USR_BLADE_DIR)/sbin
USR_BLADE_LIB_DIR       = $(USR_BLADE_DIR)/lib


BLADE_LINT_LIB_DIR= $(UTSBASE)/$(PLATFORM)/blade/lint-libs/$(OBJS_DIR)
```



```
#       Define Objects
#
BLADE_OBJS     = blade.o

#
#       Option conf file
BLADE_OPTION    = options

#
#       Include the makefiles which define build rule templates, the
#       collection of files per module, and a few specific flags. Note
#       that order is significant, just as with an include path. The
#       first build rule template which matches the files name will be
#       used. By including these in order from most machine dependent
#       to most machine independent, we allow a machine dependent file
#       to be used in preference over a machine independent version
#       (Such as a machine specific optimization, which preserves the
#       interfaces.)
#

include $(UTSTREE)/sun4u/blade/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define modules (must come after Makefile.sun4u).
#       Define modules (must come after Makefile.sun4u, for CLOSED_BUILD).
#
BLADE_KMODS             = platmod
BLADE_KMODS             += bscbus
BLADE_KMODS             += bscv

LINTS_DIR       = $(OBJS_DIR)
LINT_LIB_DIR    = $(UTSBASE)/$(PLATFORM)/blade/lint-libs/$(OBJS_DIR)
LINT_LIB= $(UTSBASE)/$(PLATFORM)/lint-libs/$(OBJS_DIR)/llib-lunix.ln
GEN_LINT_LIB= $(UTSBASE)/$(PLATFORM)/lint-libs/$(OBJS_DIR)/llib-lgenunix.ln


#
#       Define the actual specific platforms
#
MACHINE_DEFS    += -D$(PLATFORM) -D_MACHDEP -DSFMMU

#
#       Define for inline pre-processing since
#       cpp not smart about v9 yet.
#
#CPP_DEFS_32    =
#CPP_DEFS_64    = -D__sparcv9
#CPP_DEFS       = $(CPP_DEFS_$(CLASS))

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED
LINTTAGS        += -erroff=E_STATIC_UNUSED
LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
```

```
 122 LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
```

```
**********************************************************
    2644 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/grover/Makefile.grover.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# uts/sun4u/grover/Makefile.grover
# Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"
#
#       Global definitions for sun4u implementation specific modules.
#


#
#       Define directories.
#
ROOT_GROVER_DIR         = $(ROOT_PLAT_DIR)/SUNW,Sun-Blade-100
ROOT_GROVER_MOD_DIR     = $(ROOT_GROVER_DIR)/kernel

ROOT_GROVER_MISC_DIR_32 = $(ROOT_GROVER_MOD_DIR)/misc
ROOT_GROVER_MISC_DIR_64 = $(ROOT_GROVER_MISC_DIR_32)/$(SUBDIR64)
ROOT_GROVER_DRV_DIR_32  = $(ROOT_GROVER_MOD_DIR)/drv
ROOT_GROVER_DRV_DIR_64  = $(ROOT_GROVER_DRV_DIR_32)/$(SUBDIR64)

ROOT_GROVER_MISC_DIR    = $(ROOT_GROVER_MISC_DIR_$(CLASS))
ROOT_GROVER_DRV_DIR     = $(ROOT_GROVER_DRV_DIR_$(CLASS))

ROOT_PLAT_MOD_DIRS      += $(ROOT_GROVER_MOD_DIR)
ROOT_PLAT_MISC_DIRS     += $(ROOT_GROVER_MISC_DIR)

USR_GROVER_DIR          = $(USR_PLAT_DIR)/SUNW,Sun-Blade-100
USR_GROVER_LINKED_DIR   = $(USR_PLAT_DIR)/$(LINKED_PLATFORM)
USR_GROVER_INC_DIR      = $(USR_GROVER_DIR)/include
USR_GROVER_ISYS_DIR     = $(USR_GROVER_INC_DIR)/sys
USR_GROVER_SBIN_DIR     = $(USR_GROVER_DIR)/sbin
USR_GROVER_LIB_DIR      = $(USR_GROVER_DIR)/lib


GROVER_LINT_LIB_DIR= $(UTSBASE)/$(PLATFORM)/grover/lint-libs/$(OBJS_DIR)

#
```



```
# Define objects.
#
GROVER_OBJS     = grover.o

include $(UTSTREE)/sun4u/grover/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define modules (must come after Makefile.sun4u).
#       Define modules (must come after Makefile.sun4u, for CLOSED_BUILD).
#
GROVER_KMODS    = platmod
GROVER_KMODS    += grfans

$(CLOSED_BUILD)CLOSED_GROVER_KMODS      += grppm

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED
LINTTAGS        += -erroff=E_STATIC_UNUSED
LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
```

```
**********************************************************
    2446 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/javelin/Makefile.javelin.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

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

#
# Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"
#
#       This makefile contains the common definitions for the
#       sun4u Javelin system dependent modules.
#

#
# Define directories
#
ROOT_JAVELIN_DIR        = $(ROOT_PLAT_DIR)/SUNW,Ultra-250
ROOT_JAVELIN_MOD_DIR    = $(ROOT_JAVELIN_DIR)/kernel
ROOT_JAVELIN_KERN_DIR_32 = $(ROOT_JAVELIN_MOD_DIR)
ROOT_JAVELIN_KERN_DIR_64 = $(ROOT_JAVELIN_MOD_DIR)/$(SUBDIR64)
ROOT_JAVELIN_DRV_DIR_32 = $(ROOT_JAVELIN_MOD_DIR)/drv
ROOT_JAVELIN_DRV_DIR_64 = $(ROOT_JAVELIN_MOD_DIR)/drv/$(SUBDIR64)
ROOT_JAVELIN_MISC_DIR_32 = $(ROOT_JAVELIN_MOD_DIR)/misc
ROOT_JAVELIN_MISC_DIR_64 = $(ROOT_JAVELIN_MOD_DIR)/misc/$(SUBDIR64)

ROOT_JAVELIN_KERN_DIR   = $(ROOT_JAVELIN_KERN_DIR_$(CLASS))
ROOT_JAVELIN_MISC_DIR   = $(ROOT_JAVELIN_MISC_DIR_$(CLASS))
ROOT_JAVELIN_DRV_DIR    = $(ROOT_JAVELIN_DRV_DIR_$(CLASS))

ROOT_PLAT_MOD_DIRS      += $(ROOT_JAVELIN_MOD_DIR)
ROOT_PLAT_MISC_DIRS     += $(ROOT_JAVELIN_MISC_DIR)
ROOT_PLAT_MISC_DIRS_32  += $(ROOT_JAVELIN_MISC_DIR_32)
ROOT_PLAT_DRV_DIRS      = $(ROOT_JAVELIN_DRV_DIR)

USR_JAVELIN_DIR         = $(USR_PLAT_DIR)/SUNW,Ultra-250
USR_JAVELIN_INC_DIR     = $(USR_JAVELIN_DIR)/include
USR_JAVELIN_SBIN_DIR    = $(USR_JAVELIN_DIR)/sbin
USR_JAVELIN_LIB_DIR     = $(USR_JAVELIN_DIR)/lib
USR_JAVELIN_ISYS_DIR    = $(USR_JAVELIN_INC_DIR)/sys
```



```
JAVELIN_LINT_LIB_DIR    = $(UTSBASE)/$(PLATFORM)/javelin/lint-libs/$(OBJS_DIR)

#
# Define objects.
#
JAVELIN_OBJS    = javelin.o

include $(UTSTREE)/sun4u/javelin/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
# Define modules (must come after Makefile.sun4u)
# Define modules (must come after Makefile.sun4u, for CLOSED_BUILD)
#
JAVELIN_KMODS   = platmod
JAVELIN_KMODS   += envctrltwo
```

```
**********************************************************
    3908 Wed Oct 16 17:41:12 2013
new/usr/src/uts/sun4u/mpxu/Makefile.mpxu.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
#

#
#       Global definitions for sun4u implementation specific modules.
#


#
#       Define the name of this implementation.
#


#
#       Define directories.
#
ROOT_MPXU_DIR   = $(ROOT_PLAT_DIR)/SUNW,Sun-Fire-V240
ROOT_MPXU_MOD_DIR       = $(ROOT_MPXU_DIR)/kernel

ROOT_MPXU_DRV_DIR_32    = $(ROOT_MPXU_MOD_DIR)/drv
ROOT_MPXU_DRV_DIR_64    = $(ROOT_MPXU_MOD_DIR)/drv/$(SUBDIR64)
ROOT_MPXU_DRV_DIR       = $(ROOT_MPXU_DRV_DIR_$(CLASS))

ROOT_MPXU_MISC_DIR_32   = $(ROOT_MPXU_MOD_DIR)/misc
ROOT_MPXU_MISC_DIR_64   = $(ROOT_MPXU_MOD_DIR)/misc/$(SUBDIR64)
ROOT_MPXU_MISC_DIR      = $(ROOT_MPXU_MISC_DIR_$(CLASS))

ROOT_MPXU_CRYPTO_DIR_32 = $(ROOT_MPXU_MOD_DIR)/crypto
ROOT_MPXU_CRYPTO_DIR_64 = $(ROOT_MPXU_MOD_DIR)/crypto/$(SUBDIR64)
ROOT_MPXU_CRYPTO_DIR    = $(ROOT_MPXU_CRYPTO_DIR_$(CLASS))

USR_MPXU_DIR            = $(USR_PLAT_DIR)/SUNW,Sun-Fire-V240
USR_MPXU_INC_DIR        = $(USR_MPXU_DIR)/include
USR_MPXU_ISYS_DIR       = $(USR_MPXU_INC_DIR)/sys
USR_MPXU_SBIN_DIR       = $(USR_MPXU_DIR)/sbin
USR_MPXU_SBIN_PRTDIAG   = $(USR_MPXU_SBIN_DIR)/prtdiag
USR_MPXU_SBIN_FRUADM    = $(USR_MPXU_SBIN_DIR)/fruadm
USR_MPXU_LIB_DIR        = $(USR_MPXU_DIR)/lib
```



```
MPXU_LINT_LIB_DIR       = $(UTSBASE)/$(PLATFORM)/mpxu/lint-libs/$(OBJS_DIR)

#
# Links to UltraSparc III crypto modules
#
MPXU_CRYPTO_LINKS       = aes

#
#       Include the makefiles which define build rule templates, the
#       collection of files per module, and a few specific flags. Note
#       that order is significant, just as with an include path. The
#       first build rule template which matches the files name will be
#       used. By including these in order from most machine dependent
#       to most machine independent, we allow a machine dependent file
#       to be used in preference over a machine independent version
#       (Such as a machine specific optimization, which preserves the
#       interfaces.)
#

include $(UTSTREE)/sun4u/mpxu/Makefile.files
#
#       Include common rules.
#

include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define modules (must come after Makefile.sun4u).
#       Define modules (must come after Makefile.sun4u, for CLOSED_BUILD).
#
MPXU_KMODS      = tsalarm
$(CLOSED_BUILD)CLOSED_MPXU_KMODS        = platmod ntwdt

MODSTUBS_DIR    = $(UNIX_DIR)
LINTS_DIR       = $(OBJS_DIR)
LINT_LIB_DIR    = $(UTSBASE)/$(PLATFORM)/mpxu/lint-libs/$(OBJS_DIR)

#
#       Define the actual specific platforms
#
MACHINE_DEFS    = -D$(PLATFORM) -D_MACHDEP -DSFMMU -DMP

#
#       Define platform specific values
#
#MACHINE_DEFS   += -DNCPU=554
#MACHINE_DEFS   += -DMAX_UPA=1024
#MACHINE_DEFS   += -DIGN_SIZE=10
# Max IOSRAM TOC major version number supported
#MACHINE_DEFS   += -DMAX_IOSRAM_TOC_VER=0x1

#
#       Define for inline pre-processing since
#       cpp not smart about v9 yet.
#
CPP_DEFS_32   =
CPP_DEFS_64   = -D__sparcv9
CPP_DEFS      = $(CPP_DEFS_$(CLASS))

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED
```

```
123 LINTTAGS        += -erroff=E_STATIC_UNUSED
124 LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
125 LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
```

```
**********************************************************
    6094 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4u/opl/Makefile.opl.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
#
#       Global definitions for sun4u opl implementation specific modules.
#
# uts/sun4u/opl/Makefile.opl
#


#
#       Define directories.
#
#
#
ROOT_OPL_DIR    = $(ROOT_PLAT_DIR)/SUNW,SPARC-Enterprise
ROOT_OPL_MOD_DIR        = $(ROOT_OPL_DIR)/kernel
ROOT_OPL_KERN_DIR_32 = $(ROOT_OPL_MOD_DIR)
ROOT_OPL_KERN_DIR_64 = $(ROOT_OPL_MOD_DIR)/$(SUBDIR64)
ROOT_OPL_MISC_DIR_32 = $(ROOT_OPL_MOD_DIR)/misc
ROOT_OPL_MISC_DIR_64 = $(ROOT_OPL_MOD_DIR)/misc/$(SUBDIR64)
ROOT_OPL_DRV_DIR_32 = $(ROOT_OPL_MOD_DIR)/drv
ROOT_OPL_DRV_DIR_64 = $(ROOT_OPL_MOD_DIR)/drv/$(SUBDIR64)
ROOT_OPL_CPU_DIR_32 = $(ROOT_OPL_MOD_DIR)/cpu
ROOT_OPL_CPU_DIR_64 = $(ROOT_OPL_MOD_DIR)/cpu/$(SUBDIR64)
ROOT_OPL_CRYPTO_DIR_32 = $(ROOT_OPL_MOD_DIR)/crypto
ROOT_OPL_CRYPTO_DIR_64 = $(ROOT_OPL_MOD_DIR)/crypto/$(SUBDIR64)

ROOT_OPL_KERN_DIR       = $(ROOT_OPL_KERN_DIR_$(CLASS))
ROOT_OPL_MISC_DIR       = $(ROOT_OPL_MISC_DIR_$(CLASS))
ROOT_OPL_DRV_DIR        = $(ROOT_OPL_DRV_DIR_$(CLASS))
ROOT_OPL_CPU_DIR        = $(ROOT_OPL_CPU_DIR_$(CLASS))
ROOT_OPL_CRYPTO_DIR     = $(ROOT_OPL_CRYPTO_DIR_$(CLASS))

ROOT_PLAT_MOD_DIRS      += $(ROOT_OPL_MOD_DIR)
ROOT_PLAT_MISC_DIRS_32  += $(ROOT_OPL_MISC_DIR_32)

USR_OPL_DIR             = $(USR_PLAT_DIR)/SUNW,SPARC-Enterprise
USR_OPL_LIB_DIR         = $(USR_OPL_DIR)/lib
USR_OPL_SBIN_DIR        = $(USR_OPL_DIR)/sbin
```



```
USR_OPL_SBIN_PRTDIAG    = $(USR_OPL_SBIN_DIR)/prtdiag
USR_OPL_SBIN_FRUADM     = $(USR_OPL_SBIN_DIR)/fruadm
USR_OPL_INC_DIR         = $(USR_OPL_DIR)/include
USR_OPL_ISYS_DIR        = $(USR_OPL_INC_DIR)/sys

OPL_LINT_LIB_DIR        = $(UTSBASE)/$(PLATFORM)/opl/lint-libs/$(OBJS_DIR)
OPLMSU_OPTION           = options

#
#       Define modules.
#
OPL_KMODS       = platmod
OPL_KMODS       += dm2s
OPL_KMODS       += oplkmdrv
OPL_KMODS       += pcicmu
OPL_KMODS       += oplpanel
OPL_KMODS       += dr .WAIT drmach
OPL_KMODS       += oplmsu
OPL_KMODS       += mc-opl

#
#       CPU modules.
#
OPL_CPU_KMODS += olympus_c

#       Links to OPL crypto modules
#
OPL_CRYPTO_LINKS        = aes

#
#       Include the makefiles which define build rule templates, the
#       collection of files per module, and a few specific flags. Note
#       that order is significant, just as with an include path. The
#       first build rule template which matches the files name will be
#       used. By including these in order from most machine dependent
#       to most machine independent, we allow a machine dependent file
#       to be used in preference over a machine independent version
#       (Such as a machine specific optimization, which preserves the
#       interfaces.)
#
include $(UTSBASE)/sun4u/ngdr/Makefile.files
include $(UTSTREE)/sun4u/opl/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define closed modules (must come after Makefile.sun4u for CLOSED_BUILD).
#
$(CLOSED_BUILD)CLOSED_OPL_KMODS = scfd

#
#       Everybody needs to know how to build modstubs.o and to locate unix.o
#
UNIX_DIR        = $(UTSBASE)/$(PLATFORM)/opl/unix
MODSTUBS_DIR    = $(UNIX_DIR)
DSF_DIR         = $(UTSBASE)/$(PLATFORM)/opl/genassym
LINTS_DIR       = $(OBJS_DIR)
LINT_LIB_DIR    = $(UTSBASE)/$(PLATFORM)/opl/lint-libs/$(OBJS_DIR)

UNIX_O          = $(UNIX_DIR)/$(OBJS_DIR)/unix.o

LINT_LIB        = $(LINT_LIB_DIR)/llib-lunix.ln
```

```
120 #
121 #       Define the actual specific platforms
122 #
123 MACHINE_DEFS    = -D$(PLATFORM) -D_MACHDEP -DSFMMU -DMP
124 MACHINE_DEFS    += -D_CPU_SIGNATURE

126 #
127 # Maximum CPUID  =   01111 11 01 1 = 0x1FB (507)
128 # Maximum CHIPID = 1 01111 11 00 0 = 0x5F8 (1528)
129 #
130 MACHINE_DEFS    += -DNCPU=512
131 MACHINE_DEFS    += -DMAX_CPU_CHIPID=1529
132 MACHINE_DEFS    += -DMAX_UPA=256
133 MACHINE_DEFS    += -DIGN_SIZE=8
134 MACHINE_DEFS    += -DMAX_MEM_NODES=16
135 #
136 # UTSB_PHYS will enable user TSB physical access for TL>0
137 #
138 MACHINE_DEFS    += -DUTSB_PHYS
139 MACHINE_DEFS    += -D_OPL
140 MACHINE_DEFS    += -DOLYMPUS_SHARED_FTLB
141 MACHINE_DEFS    += -D_CMP_NO_ERROR_STEERING -D_HW_MEMSCRUB_SUPPORT
142 MACHINE_DEFS    += -DDO_CORELEVEL_LOADBAL
143 MACHINE_DEFS    += -DITLB_32M_256M_SUPPORT
144 #
145 # OLYMPUS C cross-call erratas.
146 #  - revision A can only deliver one xcall at a time.
147 #  - revision B can dispatch xcalls to 31 (IDSR_BN_SETS) CPUs at a time,
148 #    but it can not send more xcalls until all the pending xcalls are
149 #    dispatched. In other words, all previous 31 xcall slots must be
150 #    in non-busy state before further xcalls can be issued.
151 MACHINE_DEFS += -DOLYMPUS_C_REV_A_ERRATA_XCALL
152 MACHINE_DEFS += -DOLYMPUS_C_REV_B_ERRATA_XCALL
153 #
154 # OLYMPUS C Spurious interrupts
155 # - When an UE is detected in a interrupt packet,
156 #   Olympus-C takes an interrupt_vector_trap (TT=0x60) while
157 #   ASI_INTR_RECIEVE.BUSY is set to zero to indicate the existence
158 #   of the error. Software will see this as a spurious interrupt since
159 #   the interrupt busy bit is set to zero. SW will still need to
160 #   explicitly clear the interrupt busy bit to reset the HW state.
161 #   Failure to do so will result in the processor continuously taking
162 #   an interrupt vector trap when PSTATE.IE is reset to one.
163 #   Note that UE in interrupt packet is reported to the SP and handled
164 #   accordingly. For the domain, the system should panic as it is not
165 #   recoverable.
166 MACHINE_DEFS += -DCLEAR_INTR_BUSYBIT_ON_SPURIOUS

168 #
169 # For now, disable these lint checks; maintainers should endeavor
170 # to investigate and remove these for maximum lint coverage.
171 # Please do not carry these forward to new Makefiles.
172 #
173 LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
174 LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
175 LINTTAGS        += -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED
176 LINTTAGS        += -erroff=E_STATIC_UNUSED
177 LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
178 LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV

180 .KEEP_STATE:
```

```
**********************************************************
    2602 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4u/sunfire/Makefile.sunfire.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"
#
#       Global definitions for sun4u implementation specific modules.
#


#
#       Define the name of this implementation.
#

#
#       Define directories.
#
ROOT_SUNFIRE_DIR        = $(ROOT_PLAT_DIR)/SUNW,Ultra-Enterprise
ROOT_SUNFIRE_MOD_DIR    = $(ROOT_SUNFIRE_DIR)/kernel

ROOT_SUNFIRE_DRV_DIR_32 = $(ROOT_SUNFIRE_MOD_DIR)/drv
ROOT_SUNFIRE_DRV_DIR_64 = $(ROOT_SUNFIRE_MOD_DIR)/drv/$(SUBDIR64)
ROOT_SUNFIRE_DRV_DIR    = $(ROOT_SUNFIRE_DRV_DIR_$(CLASS))
ROOT_PLAT_DRV_DIRS      = $(ROOT_SUNFIRE_DRV_DIR)

ROOT_SUNFIRE_MISC_DIR_32        = $(ROOT_SUNFIRE_MOD_DIR)/misc
ROOT_SUNFIRE_MISC_DIR_64        = $(ROOT_SUNFIRE_MOD_DIR)/misc/$(SUBDIR64)
ROOT_SUNFIRE_MISC_DIR   = $(ROOT_SUNFIRE_MISC_DIR_$(CLASS))

USR_SUNFIRE_DIR         = $(USR_PLAT_DIR)/SUNW,Ultra-Enterprise
USR_SUNFIRE_INC_DIR     = $(USR_SUNFIRE_DIR)/include
USR_SUNFIRE_ISYS_DIR    = $(USR_SUNFIRE_INC_DIR)/sys
USR_SUNFIRE_SBIN_DIR    = $(USR_SUNFIRE_DIR)/sbin
USR_SUNFIRE_LIB_DIR     = $(USR_SUNFIRE_DIR)/lib

SUNFIRE_LINT_LIB_DIR= $(UTSBASE)/$(PLATFORM)/sunfire/lint-libs/$(OBJS_DIR)

#
```



```
# Define objects.
#
SUNFIRE_OBJS    = sunfire.o

include $(UTSTREE)/sun4u/sunfire/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define modules (must come after Makefile.sun4u).
#       Define modules (must come after Makefile.sun4u for CLOSED_BUILD).
#
SUNFIRE_KMODS                   = ac central environ fhc simmstat sysctrl sram
$(CLOSED_BUILD)CLOSED_SUNFIRE_KMODS     = platmod

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_SUPPRESSION_DIRECTIVE_UNUSED
LINTTAGS        += -erroff=E_STATIC_UNUSED
LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
```

```
**********************************************************
    3705 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4u/sys/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# uts/sun4u/sys/Makefile
#
UTSBASE = ../..

#
# include global definitions
#
include ../Makefile.sun4u

#
# Override defaults.
#
FILEMODE         = 644

SUN4_HDRS=                      \
        async.h                 \
        clock.h                 \
        cmp.h                   \
        cpc_ultra.h             \
        cpu_sgnblk_defs.h       \
        ddi_subrdefs.h          \
        dvma.h                  \
        eeprom.h                \
        errclassify.h           \
        fcode.h                 \
        fc_plat.h               \
        idprom.h                \
        intr.h                  \
        intreg.h                \
        ivintr.h                \
        memlist_plat.h          \
        memnode.h               \
        nexusdebug.h            \
        prom_debug.h            \
        scb.h                   \
```



```
        sun4asi.h               \
        tod.h                   \
        trapstat.h              \
        vis.h                   \
        vm_machparam.h          \
        x_call.h                \
        xc_impl.h               \
        zsmach.h

$(CLOSED_BUILD)CLOSED_SUN4_HDRS=        \
        memtestio.h

HDRS=   \
        cheetahregs.h           \
        cpr_impl.h              \
        cpu_impl.h              \
        ecc_kstat.h             \
        envctrl.h               \
        envctrl_gen.h           \
        envctrl_ue250.h         \
        envctrl_ue450.h         \
        gpio_87317.h            \
        iocache.h               \
        iommu.h                 \
        machasi.h               \
        machclock.h             \
        machcpuvar.h            \
        machparam.h             \
        machsystm.h             \
        machthread.h            \
        mem_cache.h             \
        mmu.h                   \
        opl_module.h            \
        prom_plat.h             \
        pte.h                   \
        sbd_ioctl.h             \
        spitregs.h              \
        starfire.h              \
        sysioerr.h              \
        sysiosbus.h             \
        todmostek.h             \
        traptrace.h

$(CLOSED_BUILD)CLOSED_HDRS=     \
        memtestio_ch.h          \
        memtestio_chp.h         \
        memtestio_ja.h          \
        memtestio_jg.h          \
        memtestio_oc.h          \
        memtestio_pn.h          \
        memtestio_sf.h          \
        memtestio_sr.h          \
        memtestio_u.h

I2CHDRS =       clients/max1617.h misc/i2c_svc.h clients/i2c_client.h   \
                clients/hpc3130.h clients/lm75.h        \
                clients/pcf8591.h clients/ssc050.h
                clients/pcf8591.h clients/ssc050.h $(CLOSED_I2CHDRS)

I2C_DIRS=       clients misc
USR_PSM_ISYS_I2C_ROOT=  $(USR_PSM_ISYS_DIR)/i2c
USR_PSM_ISYS_I2C_DIRS=  $(USR_PSM_ISYS_I2C_ROOT) \
                        $(I2C_DIRS:%=$(USR_PSM_ISYS_I2C_ROOT)/%)

ROOTI2CHDRS=            $(I2CHDRS:%=$(USR_PSM_ISYS_I2C_ROOT)/%)
```

```
110 MONHDRS=
111 #MONHDRS=       eeprom.h    idprom.h    keyboard.h  password.h

113 USR_PSM_MON_DIR=        $(USR_PSM_ISYS_DIR)/mon

115 ROOTHDRS=               $(HDRS:%=$(USR_PSM_ISYS_DIR)/%)
130 $(CLOSED_BUILD)ROOTHDRS += $(CLOSED_HDRS:%=$(USR_PSM_ISYS_DIR)/%)

117 SUN4_ROOTHDRS=          $(SUN4_HDRS:%=$(USR_PSM_ISYS_DIR)/%)
133 $(CLOSED_BUILD)SUN4_ROOTHDRS += $(CLOSED_SUN4_HDRS:%=$(USR_PSM_ISYS_DIR)/%)

119 ROOTMONHDRS=            $(MONHDRS:%=$(USR_PSM_MON_DIR)/%)

121 ROOTDIR=                $(ROOT)/usr/share/src
122 ROOTDIRS=               $(ROOTDIR)/uts $(ROOTDIR)/uts/$(PLATFORM)

124 ROOTLINK=               $(ROOTDIR)/uts/$(PLATFORM)/sys
125 LINKDEST=               ../../../../platform/$(PLATFORM)/include/sys

127 CHECKHDRS=              $(HDRS:%.h=%.check) \
128                         $(MONHDRS:%.h=mon/%.check) \
129                         $(SUN4_HDRS:%.h=%.cmncheck)

147 $(CLOSED_BUILD)CHECKHDRS += $(CLOSED_HDRS:%.h=$(CLOSED)/uts/sun4u/sys/%.check)
148 $(CLOSED_BUILD)CHECKHDRS += \
149                 $(CLOSED_I2CHDRS:%.h=$(CLOSED)/uts/sun4u/sys/i2c/%.check)

131 .KEEP_STATE:

133 .PARALLEL: $(CHECKHDRS) $(ROOTHDRS) $(ROOTMONHDRS) $(SUN4_ROOTHDRS)

135 install_h: $(ROOTDIRS) $(USR_PSM_ISYS_I2C_DIRS) .WAIT \
136                 $(ROOTHDRS) $(ROOTI2CHDRS) \
137                 $(ROOTMONHDRS) \
138                 $(SUN4_ROOTHDRS) $(ROOTLINK)

140 check:  $(CHECKHDRS)

142 #
143 # install rules
144 #
145 $(USR_PSM_MON_DIR):     $(USR_PSM_ISYS_DIR)
146         $(INS.dir)

148 $(USR_PSM_ISYS_I2C_DIRS):
149         $(INS.dir)

171 $(USR_PSM_ISYS_DIR)/%:  $(CLOSED)/uts/sun4u/sys/% $(USR_PSM_ISYS_DIR)
172         $(INS.file)

174 $(USR_PSM_ISYS_DIR)/%:  $(CLOSED)/uts/sun4/sys/% $(USR_PSM_ISYS_DIR)
175         $(INS.file)

151 $(USR_PSM_ISYS_DIR)/%:  ../../sfmmu/sys/% $(USR_PSM_ISYS_DIR)
152         $(INS.file)

154 $(USR_PSM_ISYS_DIR)/%:  ../../sun4/sys/% $(USR_PSM_ISYS_DIR)
155         $(INS.file)

157 $(USR_PSM_MON_DIR)/%:   mon/% $(USR_PSM_MON_DIR)
158         $(INS.file)

160 $(ROOTDIRS):
161         $(INS.dir)

163 # -r because this used to be a directory and is now a link.
```



```
164 $(ROOTLINK):    $(ROOTDIRS)
165         -$(RM) -r $@; $(SYMLINK) $(LINKDEST) $@

167 mon/%.check:    mon/%.h
168         $(DOT_H_CHECK)

170 %.check:        ../../sfmmu/sys/%.h
171         $(DOT_H_CHECK)
172 %.cmncheck:     ../../sun4/sys/%.h
173         $(DOT_H_CHECK)

175 FRC:

177 include ../../Makefile.targ
```

```
**********************************************************
    2766 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4u/tazmo/Makefile.tazmo.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# uts/sun4u/tazmo/Makefile.impl
# Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"
#
#       Global definitions for sun4u implementation specific modules.
#


#
#       Define directories.
#
ROOT_TAZMO_DIR          = $(ROOT_PLAT_DIR)/SUNW,Ultra-4
ROOT_TAZMO_MOD_DIR      = $(ROOT_TAZMO_DIR)/kernel

ROOT_TAZMO_KERN_DIR_32  = $(ROOT_TAZMO_MOD_DIR)
ROOT_TAZMO_KERN_DIR_64  = $(ROOT_TAZMO_MOD_DIR)/$(SUBDIR64)
ROOT_TAZMO_DRV_DIR_32   = $(ROOT_TAZMO_MOD_DIR)/drv
ROOT_TAZMO_DRV_DIR_64   = $(ROOT_TAZMO_MOD_DIR)/drv/$(SUBDIR64)
ROOT_TAZMO_MISC_DIR_32  = $(ROOT_TAZMO_MOD_DIR)/misc
ROOT_TAZMO_MISC_DIR_64  = $(ROOT_TAZMO_MOD_DIR)/misc/$(SUBDIR64)

ROOT_TAZMO_KERN_DIR     = $(ROOT_TAZMO_KERN_DIR_$(CLASS))
ROOT_TAZMO_MISC_DIR     = $(ROOT_TAZMO_MISC_DIR_$(CLASS))
ROOT_TAZMO_DRV_DIR      = $(ROOT_TAZMO_DRV_DIR_$(CLASS))

ROOT_PLAT_MOD_DIRS      += $(ROOT_TAZMO_MOD_DIR)
ROOT_PLAT_MISC_DIRS     += $(ROOT_TAZMO_MISC_DIR)
ROOT_PLAT_MISC_DIRS_32  += $(ROOT_TAZMO_MISC_DIR_32)
ROOT_PLAT_DRV_DIRS      = $(ROOT_TAZMO_DRV_DIR)

USR_TAZMO_DIR           = $(USR_PLAT_DIR)/SUNW,Ultra-4
USR_TAZMO_LINKED_DIR    = $(USR_PLAT_DIR)/$(LINKED_PLATFORM)
USR_TAZMO_INC_DIR       = $(USR_TAZMO_DIR)/include
USR_TAZMO_ISYS_DIR      = $(USR_TAZMO_INC_DIR)/sys
USR_TAZMO_SBIN_DIR      = $(USR_TAZMO_DIR)/sbin
USR_TAZMO_LIB_DIR       = $(USR_TAZMO_DIR)/lib
```



```
TAZMO_LINT_LIB_DIR= $(UTSBASE)/$(PLATFORM)/tazmo/lint-libs/$(OBJS_DIR)

#
# Define objects.
#
TAZMO_OBJS      = tazmo.o

include $(UTSBASE)/sun4u/tazmo/Makefile.files

#
#       Include common rules.
#
include $(UTSTREE)/sun4u/Makefile.sun4u

#
#       Define modules (must come after Makefile.sun4u).
#       Define modules (must come after Makefile.sun4u, for CLOSED_BUILD).
#
TAZMO_KMODS     = platmod
TAZMO_KMODS     += envctrl

#
# For now, disable these lint checks; maintainers should endeavor
# to investigate and remove these for maximum lint coverage.
# Please do not carry these forward to new Makefiles.
#
LINTTAGS        += -erroff=E_PTRDIFF_OVERFLOW
LINTTAGS        += -erroff=E_BAD_PTR_CAST_ALIGN
LINTTAGS        += -erroff=E_ASSIGN_NARROW_CONV
LINTTAGS        += -erroff=E_SUSPICIOUS_COMPARISON
```

```
**********************************************************
    7946 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4v/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#       This makefile drives the production of all implementation architecture
#       dependent modules for the sun4v architecture.
#

UTSBASE = ..

include Makefile.sun4v
include Makefile.stpaul
include Makefile.huron
include Makefile.maramba
include Makefile.thunder
include Makefile.turgo
include Makefile.congo
include Makefile.monza

USR_GLENDALE_DIR        = $(USR_PLAT_DIR)/SUNW,Sun-Blade-T6320
USR_GLENDALE_SBIN_DIR   = $(USR_GLENDALE_DIR)/sbin
USR_GLENDALE_LIB_DIR    = $(USR_GLENDALE_DIR)/lib


#
#       The following are SPARC specific (rather than sun4v) specific modules
#       which are required for the sun4v kernel to completely lint. They are
#       not involved in the build in any other way. In order to minimize
#       build time, it is assumed that they are up to date. But since sun4v
#       is really a separate architecture we cannot use the v7 sparc modules.
#
SPARC_LIB_DIR    = $(UTSBASE)/sparc/lint-libs/$(OBJS_DIR)

SPARC_LINTS      =

#
#
#
```



```
LINT_LIBS        = $(LINT_LIB) \
                   $(LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(CLOSED_LINT_KMODS:%=$(LINT_LIB_DIR)/llib-l%.ln) \
                   $(SPARC_LINTS:%=$(SPARC_LIB_DIR)/llib-l%.ln)

def             :=      TARGET= def
all             :=      TARGET= all
install         :=      TARGET= install
install_h       :=      TARGET= install_h
clean           :=      TARGET= clean
clobber         :=      TARGET= clobber
lint            :=      TARGET= lint
lintlib         :=      TARGET= lintlib
modlintlib      :=      TARGET= modlintlib
modlist         :=      TARGET= modlist
modlist modlist.sparc :=        NO_STATE= -K $$MODSTATE$$$$
clean.lint      :=      TARGET= clean.lint
check           :=      TARGET= check

.KEEP_STATE:

.PARALLEL:      $(PARALLEL_KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
                modlist modlist.sparc

# Override for CPU_KMODS... they cannot be built
# in parallel
.NO_PARALLEL:   $(CPU_KMODS)

def all clean clobber clean.lint: genassym unix .WAIT \
        $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) $(IMPLEMENTATIONS)

# list the modules under sun4v.
modlist: unix $(KMODS) $(CLOSED_KMODS) $(XMODS) $(CLOSED_XMODS) \
        $(IMPLEMENTATIONS)

# list the modules for Install -k sun4v.
modlist.karch: modlist modlist.sparc

modlist.sparc:
        @cd $(SRC)/uts/sparc; pwd; $(MAKE) $(NO_STATE) modlist

install: install_platforms genassym unix .WAIT $(KMODS) $(CLOSED_KMODS) \
        $(XMODS) $(CLOSED_XMODS) $(IMPLEMENTATIONS)

lintlib:        unix

modlintlib:     $(LINT_KMODS) $(CLOSED_LINT_KMODS)

genassym unix $(KMODS): FRC
        @cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

$(IMPLEMENTATIONS):     FRC
        @cd $@; pwd; THISIMPL=$@ $(MAKE) $(NO_STATE) $(TARGET)

$(XMODS):       FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(NO_STATE) $(TARGET); \
        else \
                true; \
        fi

$(CLOSED_XMODS):        FRC
        @if [ -f $(CLOSED)/uts/sun4v/$@/Makefile  ]; then \
                cd $(CLOSED)/uts/sun4v/$@; pwd; \
                    $(MAKE) $(NO_STATE) $(TARGET); \
        else \
```

```
125                 true; \
126         fi

128 $(CLOSED_KMODS):        FRC
129         cd $(CLOSED)/uts/sun4v/$@; pwd; $(MAKE) $(NO_STATE) $(TARGET)

131 install_h check:        install_platforms $(IMPLEMENTATIONS) FRC
132         @cd sys; pwd; $(MAKE) $(TARGET)
133         @cd vm; pwd; $(MAKE) $(TARGET)

135 #
136 # Rules for the /platforms directories.  This is hardwired here because
137 # the first stage of the project (KBI) only implements the userland
138 # changes, but the only reasonable place to record the aliases is
139 # here in kernel land.
140 #
141 $(ROOT_PLAT_DIRS): $(ROOT_PLAT_DIR)
142         -$(INS.dir)

144 $(LINKED_PLATFORMS:%=$(ROOT_PLAT_DIR)/%): $(ROOT_PLAT_DIR)
145         $(INS.slink1)

147 #
148 # create directories in /usr/platform/ for the implementations that are
149 # defined in $(IMPLEMENTED_PLATFORM)
150 #

152 # Foreach $(IMPLEMENTED_PLATFORM) there can be a list of $(LINKED_PLATFORMS)
153 # that are linked to it.
154 #
155 $(USR_PLAT_DIR)/$(IMPLEMENTED_PLATFORM):        $(USR_PLAT_DIR)
156         -$(INS.dir)

158 #
159 # create the links in /usr/platform/ foreach $(LINKED_PLATFORMS)
160 # to it's corresponding $(IMPLEMENTED_PLATFORM).
161 #
162 PLATFORMS       = $(LINKED_PLATFORMS)

164 $(USR_PLAT_DIRS): $(USR_PLAT_DIR)
165         $(INS.slink3)

167 PLATFORMS       += $(IMPLEMENTED_PLATFORM)


170 #
171 # Make the /platforms directories.  This is hardwired here because
172 # the first stage of the project (KBI) only implements the userland
173 # changes, but the only reasonable place to record the aliases is
174 # here in kernel land.
175 #

177 install_platforms:      $(ROOT_PSM_DIR) $(USR_PSM_DIR) \
178                         $(ROOT_PLAT_DIRS) $(USR_PLAT_DIRS) \
179                         $(LINKED_PLATFORMS:%=$(ROOT_PLAT_DIR)/%) \
180                         $(USR_DESKTOP_DIR) $(USR_DESKTOP_INC_DIR) \
181                         $(USR_DESKTOP_SBIN_DIR) $(USR_DESKTOP_LIB_DIR) \
182                         $(USR_STPAUL_DIR) $(USR_STPAUL_SBIN_DIR) \
183                         $(USR_STPAUL_LIB_DIR) \
184                         $(USR_GLENDALE_DIR) $(USR_GLENDALE_SBIN_DIR) \
185                         $(USR_GLENDALE_LIB_DIR) \
186                         $(USR_HURON_DIR) \
187                         $(USR_HURON_SBIN_DIR) $(USR_HURON_LIB_DIR) \
188                         $(USR_MARAMBA_DIR) $(USR_MARAMBA_SBIN_DIR) \
189                         $(USR_MARAMBA_LIB_DIR) \
190                         $(USR_THUNDER_DIR) $(USR_THUNDER_SBIN_DIR) \
```



```
191                         $(USR_THUNDER_LIB_DIR) \
192                         $(USR_TURGO_DIR) $(USR_TURGO_SBIN_DIR) \
193                         $(USR_TURGO_LIB_DIR) \
194                         $(USR_CONGO_DIR) $(USR_CONGO_SBIN_DIR) \
195                         $(USR_CONGO_LIB_DIR) \
196                         $(USR_MONZA_DIR) \
197                         $(USR_MONZA_SBIN_DIR) $(USR_MONZA_SBIN_LINKS)


200 #
201 # rules for making include, sbin, lib dirs/links in
202 # /usr/platform/$(PLATFORM)/ for desktop platforms
203 #
204 $(USR_DESKTOP_INC_DIR):         $(USR_DESKTOP_DIR)
205         $(INS.slink4)

207 $(USR_DESKTOP_SBIN_DIR):        $(USR_DESKTOP_DIR)
208         $(INS.slink5)

210 $(USR_DESKTOP_LIB_DIR):         $(USR_DESKTOP_DIR)
211         -$(INS.dir)

213 $(USR_STPAUL_DIR):              $(USR_SUN4V_PLAT_DIR)
214         -$(INS.dir)

216 $(USR_STPAUL_SBIN_DIR):         $(USR_STPAUL_DIR)
217         $(INS.slink5)

219 $(USR_STPAUL_LIB_DIR):          $(USR_STPAUL_DIR)
220         -$(INS.dir)

222 $(USR_HURON_DIR):               $(USR_SUN4V_PLAT_DIR)
223         -$(INS.dir)

225 $(USR_HURON_SBIN_DIR):          $(USR_HURON_DIR)
226         $(INS.slink5)

228 $(USR_HURON_LIB_DIR):           $(USR_HURON_DIR)
229         -$(INS.dir)

231 $(USR_GLENDALE_DIR):            $(USR_SUN4V_PLAT_DIR)
232         -$(INS.dir)

234 $(USR_GLENDALE_SBIN_DIR):               $(USR_GLENDALE_DIR)
235         $(INS.slink5)

237 $(USR_GLENDALE_LIB_DIR):               $(USR_GLENDALE_DIR)
238         -$(INS.dir)

240 $(USR_MARAMBA_DIR):             $(USR_SUN4V_PLAT_DIR)
241         -$(INS.dir)

243 $(USR_MARAMBA_SBIN_DIR):        $(USR_MARAMBA_DIR)
244         $(INS.slink5)

246 $(USR_MARAMBA_LIB_DIR):         $(USR_MARAMBA_DIR)
247         -$(INS.dir)

249 $(USR_THUNDER_DIR):             $(USR_SUN4V_PLAT_DIR)
250         -$(INS.dir)

252 $(USR_THUNDER_SBIN_DIR):        $(USR_THUNDER_DIR)
253         $(INS.slink5)

255 $(USR_THUNDER_LIB_DIR):         $(USR_THUNDER_DIR)
256         -$(INS.dir)
```

```
$(USR_TURGO_DIR):               $(USR_SUN4V_PLAT_DIR)
        -$(INS.dir)

$(USR_TURGO_SBIN_DIR):          $(USR_TURGO_DIR)
        $(INS.slink5)

$(USR_TURGO_LIB_DIR):           $(USR_TURGO_DIR)
        -$(INS.dir)

$(USR_CONGO_DIR):               $(USR_SUN4V_PLAT_DIR)
        -$(INS.dir)

$(USR_CONGO_SBIN_DIR):          $(USR_CONGO_DIR)
        $(INS.slink5)

$(USR_CONGO_LIB_DIR):           $(USR_CONGO_DIR)
        -$(INS.dir)

$(USR_MONZA_DIR):               $(USR_SUN4V_PLAT_DIR)
        -$(INS.dir)

$(USR_MONZA_SBIN_DIR):          $(USR_MONZA_DIR)
        -$(INS.dir)

$(USR_MONZA_SBIN_LINKS):        $(USR_MONZA_SBIN_DIR)
        $(INS.slink7)

#
#       Full kernel lint target.
#
LINT_TARGET     = globallint

globallint:
        @-$(ECHO) "\nSUN4V KERNEL: global crosschecks:"
        @-$(LINT) $(LINTFLAGS) $(LINT_LIBS) 2>&1 | $(LGREP.2)

lint:   lintlib .WAIT modlintlib .WAIT $(SPARC_LINTS) $(LINT_DEPS) \
        $(IMPLEMENTATIONS) $(LINT_CPU_KMODS)

include ../Makefile.targ

#
# Cross-reference customization: build a cross-reference over all of the
# sun4v-related directories.
#
XRDIRS  = ../sun4v ../sun4 ../sfmmu ../sparc ../sun ../common
SHARED_XRDIRS   = ../sun4v ../sun4 ../sfmmu ../sparc ../sun ../common
CLOSED_XRDIRS   = $(SHARED_XRDIRS:../%=../% ../../../closed/uts/%)
XRDIRS          = $(SHARED_XRDIRS)
$(CLOSED_BUILD)XRDIRS   = $(CLOSED_XRDIRS:../../../closed/uts/sfmmu=)

XRPRUNE = i86pc sun4u intel

cscope.out tags: FRC
        $(XREF) -x $@
```

```
**********************************************************
   12121 Wed Oct 16 17:41:13 2013
new/usr/src/uts/sun4v/Makefile.sun4v.shared
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
```

```
   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 2010 Sun Microsystems, Inc.  All rights reserved.
  24 # Use is subject to license terms.
  25 # Copyright (c) 2013 Andrew Stormont.  All rights reserved.
  26 #
  27 #       This makefile contains the common definitions for the sun4v unix
  28 #       and all sun4v implementation architecture dependent modules.
  29 #

  31 #
  32 #       Machine type (implementation architecture):
  33 #
  34 PLATFORM         = sun4v
  35 LINKED_PLATFORMS += SUNW,Sun-Fire-T1000
  36 LINKED_PLATFORMS += SUNW,SPARC-Enterprise-T5120
  37 LINKED_PLATFORMS += SUNW,SPARC-Enterprise-T5220
  38 LINKED_PLATFORMS += SUNW,T5140
  39 LINKED_PLATFORMS += SUNW,T5240
  40 LINKED_PLATFORMS += SUNW,T5440
  41 LINKED_PLATFORMS += SUNW,SPARC-Enterprise-T1000
  42 LINKED_PLATFORMS += SUNW,Sun-Blade-T6300
  43 LINKED_PLATFORMS += SUNW,Sun-Blade-T6320
  44 LINKED_PLATFORMS += SUNW,Netra-CP3260
  45 LINKED_PLATFORMS += SUNW,Netra-T5220
  46 LINKED_PLATFORMS += SUNW,USBRDT-5240
  47 LINKED_PLATFORMS += SUNW,Netra-T5440
  48 LINKED_PLATFORMS += SUNW,Sun-Blade-T6340
  49 PROMIF           = ieee1275
  50 PSMBASE          = $(UTSBASE)/../psm

  52 #
  53 #       uname -m value
  54 #
  55 UNAME_M         = $(PLATFORM)

  57 #
  58 # Definitions for the platform-specific /platform directories.
```



```
  59 #
  60 # PLATFORMS designates those sun4v machines which have no platform
  61 # specific code.
  62 #
  63 # IMPLEMENTATIONS is used to designate sun4v machines which have
  64 # platform specific modules.  All code specific to a given implementation
  65 # resides in the appropriately named subdirectory.   This requires
  66 # these platforms to have their own Makefiles to define ROOT_PLAT_DIRS,
  67 # USR_PLAT_DIRS, etc.
  68 # The number of IMPLEMENTATIONS should not grow!
  69 #
  70 # So if we had an implementation named 'foo', we would need the following
  71 # Makefiles in the foo subdirectory:
  72 #
  73 #       sun4v/foo/Makefile
  74 #       sun4v/foo/Makefile.foo
  75 #       sun4v/foo/Makefile.targ
  76 #

  78 #
  79 # all PLATFORMS that do not belong in the $(IMPLEMENTATIONS) list.
  80 # This list should be empty.  A platform without platform modules
  81 # is a plain, generic sun4v platform.
  82 #
  83 #IMPLEMENTED_PLATFORM   =
  84 #PLATFORMS                = $(IMPLEMENTED_PLATFORM)

  86 IMPLEMENTATIONS         = ontario montoya huron maramba

  88 #ROOT_PLAT_DIRS         = $(PLATFORMS:%=$(ROOT_PLAT_DIR)/%)
  89 #USR_PLAT_DIRS          = $(PLATFORMS:%=$(USR_PLAT_DIR)/%)

  91 #USR_DESKTOP_DIR                = $(USR_PLAT_DIR)/$(IMPLEMENTED_PLATFORM)
  92 #USR_DESKTOP_INC_DIR    = $(USR_DESKTOP_DIR)/include
  93 #USR_DESKTOP_SBIN_DIR   = $(USR_DESKTOP_DIR)/sbin
  94 #USR_DESKTOP_LIB_DIR    = $(USR_DESKTOP_DIR)/lib

  96 #
  97 #       Define supported builds
  98 #
  99 DEF_BUILDS      = $(DEF_BUILDS64)
 100 ALL_BUILDS      = $(ALL_BUILDS64)

 102 #
 103 #       Everybody needs to know how to build modstubs.o and to locate unix.o
 104 #
 105 UNIX_DIR         = $(UTSBASE)/$(PLATFORM)/unix
 106 GENLIB_DIR       = $(UTSBASE)/$(PLATFORM)/genunix
 107 MODSTUBS_DIR     = $(UNIX_DIR)
 108 DSF_DIR          = $(UTSBASE)/$(PLATFORM)/genassym
 109 LINTS_DIR        = $(OBJS_DIR)
 110 LINT_LIB_DIR     = $(UTSBASE)/$(PLATFORM)/lint-libs/$(OBJS_DIR)

 112 DTRACESTUBS_O    = $(OBJS_DIR)/dtracestubs.o
 113 DTRACESTUBS      = $(OBJS_DIR)/libdtracestubs.so

 115 UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
 116 MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
 117 GENLIB           = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so

 119 LINT_LIB         = $(LINT_LIB_DIR)/llib-lunix.ln
 120 GEN_LINT_LIB     = $(LINT_LIB_DIR)/llib-lgenunix.ln

 122 LINT64_DIRS      = $(LINT64_BUILDS:%=$(UTSBASE)/$(PLATFORM)/lint-libs/%)
 123 LINT64_FILES     = $(LINT64_DIRS:%=%/llib-l$(MODULE).ln)
```

```
125 #
126 #       cpu and platform modules need to know how to build their own symcheck mo
127 #
128 PLATMOD          = platmod
129 PLATLIB          = $(PLAT_DIR)/$(OBJS_DIR)/libplatmod.so

131 CPUNAME          = cpu
132 CPULIB           = $(CPU_DIR)/$(OBJS_DIR)/libcpu.so

134 SYM_MOD          = $(OBJS_DIR)/unix.sym

136 #
137 #       Include the makefiles which define build rule templates, the
138 #       collection of files per module, and a few specific flags. Note
139 #       that order is significant, just as with an include path. The
140 #       first build rule template which matches the files name will be
141 #       used. By including these in order from most machine dependent
142 #       to most machine independent, we allow a machine dependent file
143 #       to be used in preference over a machine independent version
144 #       (Such as a machine specific optimization, which preserves the
145 #       interfaces.)
146 #
147 include $(UTSBASE)/sun4/Makefile.files
148 include $(UTSTREE)/$(PLATFORM)/Makefile.files
149 include $(UTSBASE)/sfmmu/Makefile.files
150 include $(UTSBASE)/sparc/v9/Makefile.files
151 include $(UTSBASE)/sparc/Makefile.files
152 include $(UTSTREE)/sun/Makefile.files
153 include $(SRC)/psm/promif/$(PROMIF)/common/Makefile.files
154 include $(SRC)/psm/promif/$(PROMIF)/$(PLATFORM)/Makefile.files
155 include $(UTSTREE)/common/Makefile.files

157 #
158 #       Include machine independent rules. Note that this does not imply
159 #       that the resulting module from rules in Makefile.uts is machine
160 #       independent. Only that the build rules are machine independent.
161 #
162 include $(UTSBASE)/Makefile.uts

164 CTFMERGE_GUDIR  = sun4v

166 #
167 #       machine specific optimization, override default in Makefile.master
168 #
169 CC_XARCH        = -m64 -xarch=sparcvis
170 AS_XARCH        = -xarch=v9v
171 COPTIMIZE       = -xO3
172 CCMODE          = -Xa

174 CFLAGS          = -xchip=ultra $(CCABS32) $(CCREGSYM)
175 CFLAGS          += $(CC_XARCH)
176 CFLAGS          += $(COPTIMIZE)
177 CFLAGS          += $(EXTRA_CFLAGS)
178 CFLAGS          += $(XAOPT)
179 CFLAGS          += $(INLINES) -D_ASM_INLINES
180 CFLAGS          += $(CCMODE)
181 CFLAGS          += $(SPACEFLAG)
182 CFLAGS          += $(CERRWARN)
183 CFLAGS          += $(CTF_FLAGS_$(CLASS))
184 CFLAGS          += $(C99MODE)
185 CFLAGS          += $(CCUNBOUND)
186 CFLAGS          += $(CCNOAUTOINLINE)
187 CFLAGS          += $(CCSTATICSYM)
188 CFLAGS          += $(CC32BITCALLERS)
189 CFLAGS          += $(IROPTFLAG)
190 CFLAGS          += $(CGLOBALSTATIC)
```



```
191 CFLAGS          += -xregs=no%float
192 CFLAGS          += -xstrconst
193 CFLAGS          += $(CSOURCEDEBUGFLAGS)
194 CFLAGS          += $(CUSERFLAGS)

196 CPPFLAGS        += -DGLREG

198 ASFLAGS         += $(AS_XARCH) -DGLREG

200 AS_INC_PATH     += -I$(DSF_DIR)/$(OBJS_DIR)

202 LINT_KMODS      += $(GENUNIX_KMODS)

204 LINT_DEFS       = -m64

206 #
207 #       The following must be defined for all implementations:
208 #
209 #       MAPFILE:                ld mapfile for the build of kernel/unix.
210 #       MODSTUBS:               Module stubs source file.
211 #       GENCONST_SRC:           genconst.c
212 #       OFFSETS:                offsets.in
213 #       PLATFORM_OFFSETS:       Platform specific mach_offsets.in
214 #       FDOFFSETS:              fd_offsets.in
215 #
216 MAPFILE          = $(UTSBASE)/sun4/conf/Mapfile
217 MODSTUBS         = $(UTSBASE)/sparc/ml/modstubs.s
218 GENCONST_SRC     = $(UTSBASE)/sun4/ml/genconst.c
219 OFFSETS          = $(UTSBASE)/sun4/ml/offsets.in
220 PLATFORM_OFFSETS = $(UTSBASE)/sun4v/ml/mach_offsets.in
221 FDOFFSETS        = $(UTSBASE)/sun/io/fd_offsets.in

223 #
224 #       Define the actual specific platforms
225 #

227 MACHINE_DEFS    = -D$(PLATFORM) -D_MACHDEP -DSFMMU
228 MACHINE_DEFS    += -DMAX_MEM_NODES=8

230 #
231 #       Software workarounds for hardware "features"
232 #

234 include $(UTSBASE)/$(PLATFORM)/Makefile.workarounds

236 #
237 #       Debugging level
238 #
239 #       Special knowledge of which special debugging options effect which
240 #       file is used to optimize the build if these flags are changed.
241 #
242 #       XXX: The above could possibly be done for more flags and files, but
243 #            is left as an experiment to the interested reader. Be forewarned,
244 #            that excessive use could lead to maintenance difficulties.
245 #
246 #       Note: kslice can be enabled for the sun4v, but is disabled by default
247 #             in all cases.
248 #

250 DEBUG_DEFS_OBJ64        =
251 DEBUG_DEFS_DBG64        = -DDEBUG
252 DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

254 DEBUG_COND_OBJ64        = $(POUND_SIGN)
255 DEBUG_COND_DBG64        =
256 IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/
```

```
258 $(IF_DEBUG_OBJ)trap.o           :=      DEBUG_DEFS      += -DTRAPDEBUG
259 $(IF_DEBUG_OBJ)mach_trap.o      :=      DEBUG_DEFS      += -DTRAPDEBUG
260 $(IF_DEBUG_OBJ)syscall_trap.o   :=      DEBUG_DEFS      += -DSYSCALLTRACE
261 $(IF_DEBUG_OBJ)clock.o          :=      DEBUG_DEFS      += -DKSLICE=0

263 IF_TRAPTRACE_OBJ = $(IF_DEBUG_OBJ)
264 # comment this out for a non-debug kernel with TRAPTRACE
265 #IF_TRAPTRACE_OBJ = $(OBJS_DIR)/

267 $(IF_TRAPTRACE_OBJ)mach_locore.o        :=      DEBUG_DEFS      += -DTRAPTRACE
268 $(IF_TRAPTRACE_OBJ)mlsetup.o            :=      DEBUG_DEFS      += -DTRAPTRACE
269 $(IF_TRAPTRACE_OBJ)syscall_trap.o       :=      DEBUG_DEFS      += -DTRAPTRACE
270 $(IF_TRAPTRACE_OBJ)startup.o            :=      DEBUG_DEFS      += -DTRAPTRACE
271 $(IF_TRAPTRACE_OBJ)mach_startup.o       :=      DEBUG_DEFS      += -DTRAPTRACE
272 $(IF_TRAPTRACE_OBJ)mp_startup.o         :=      DEBUG_DEFS      += -DTRAPTRACE
273 $(IF_TRAPTRACE_OBJ)cpu_states.o         :=      DEBUG_DEFS      += -DTRAPTRACE
274 $(IF_TRAPTRACE_OBJ)mach_cpu_states.o    :=      DEBUG_DEFS      += -DTRAPTRACE
275 $(IF_TRAPTRACE_OBJ)interrupt.o          :=      DEBUG_DEFS      += -DTRAPTRACE
276 $(IF_TRAPTRACE_OBJ)mach_interrupt.o     :=      DEBUG_DEFS      += -DTRAPTRACE
277 $(IF_TRAPTRACE_OBJ)sfmmu_asm.o          :=      DEBUG_DEFS      += -DTRAPTRACE
278 $(IF_TRAPTRACE_OBJ)trap_table.o         :=      DEBUG_DEFS      += -DTRAPTRACE
279 $(IF_TRAPTRACE_OBJ)xc.o                 :=      DEBUG_DEFS      += -DTRAPTRACE
280 $(IF_TRAPTRACE_OBJ)mach_xc.o            :=      DEBUG_DEFS      += -DTRAPTRACE
281 $(IF_TRAPTRACE_OBJ)wbuf.o               :=      DEBUG_DEFS      += -DTRAPTRACE
282 $(IF_TRAPTRACE_OBJ)trap.o               :=      DEBUG_DEFS      += -DTRAPTRACE
283 $(IF_TRAPTRACE_OBJ)mach_trap.o          :=      DEBUG_DEFS      += -DTRAPTRACE
284 $(IF_TRAPTRACE_OBJ)x_call.o             :=      DEBUG_DEFS      += -DTRAPTRACE

286 # Comment these out if you don’t want dispatcher lock statistics.

288 #$(IF_DEBUG_OBJ)lock_prim.o     := DEBUG_DEFS   += -DDISP_LOCK_STATS
289 #$(IF_DEBUG_OBJ)disp.o          := DEBUG_DEFS   += -DDISP_LOCK_STATS

291 # Comment these out if you don’t want dispatcher debugging

293 #$(IF_DEBUG_OBJ)lock_prim.o     := DEBUG_DEFS   += -DDISP_DEBUG

295 #
296 #       Collect the preprocessor definitions to be associated with *all*
297 #       files.
298 #
299 ALL_DEFS         = $(MACHINE_DEFS) $(WORKAROUND_DEFS) $(DEBUG_DEFS) \
300                    $(OPTION_DEFS)
301 GENCONST_DEFS    = $(MACHINE_DEFS) $(OPTION_DEFS)

303 #
304 # ----- TRANSITIONAL SECTION --------------------------------------------------
305 #

307 #
308 #       Not everything which *should* be a module is a module yet. The
309 #       following is a list of such objects which are currently part of
310 #       the base kernel but should soon become kmods.
311 #
312 MACH_NOT_YET_KMODS       = $(AUTOCONF_OBJS)

314 #
315 # ----- END OF TRANSITIONAL SECTION -------------------------------------------
316 #

318 #
319 #       The kernels modules which are "implementation architecture"
320 #       specific for this machine are enumerated below. Note that most
321 #       of these modules must exist (in one form or another) for each
322 #       architecture.
```



```
323 #
324 #       Common Drivers (usually pseudo drivers) (/kernel/drv):
325 #

327 #
328 #       Machine Specific Driver Modules (/kernel/drv):
329 #
330 DRV_KMODS       += bge
331 DRV_KMODS       += cnex
332 DRV_KMODS       += cpc
333 DRV_KMODS       += drctl
334 DRV_KMODS       += ds_pri
335 DRV_KMODS       += ds_snmp
336 DRV_KMODS       += ebus
337 DRV_KMODS       += fpc
338 DRV_KMODS       += glvc
339 DRV_KMODS       += mdesc
340 DRV_KMODS       += niumx
341 DRV_KMODS       += ntwdt
342 DRV_KMODS       += nxge
343 DRV_KMODS       += n2piupc
344 DRV_KMODS       += iospc
345 DRV_KMODS       += n2rng
346 DRV_KMODS       += px
347 DRV_KMODS       += qcn
348 DRV_KMODS       += rootnex
349 DRV_KMODS       += su
350 DRV_KMODS       += tpm
351 DRV_KMODS       += trapstat
352 DRV_KMODS       += vcc
353 DRV_KMODS       += vdc
354 DRV_KMODS       += vds
355 DRV_KMODS       += vldc
356 DRV_KMODS       += vlds
357 DRV_KMODS       += vnet
358 DRV_KMODS       += vnex
359 DRV_KMODS       += vsw

361 $(CLOSED_BUILD)CLOSED_DRV_KMODS += bmc
362 $(CLOSED_BUILD)CLOSED_DRV_KMODS += memtest
363 $(CLOSED_BUILD)CLOSED_DRV_KMODS += ncp
364 $(CLOSED_BUILD)CLOSED_DRV_KMODS += n2cp

361 #
362 #       Exec Class Modules (/kernel/exec):
363 #
364 EXEC_KMODS      +=

366 #
367 #       Scheduling Class Modules (/kernel/sched):
368 #
369 SCHED_KMODS     +=

371 #
372 #       File System Modules (/kernel/fs):
373 #
374 FS_KMODS        +=

376 #
377 #       Streams Modules (/kernel/strmod):
378 #
379 # STRMOD_KMODS  += kb

381 #
382 #       ’System’ Modules (/kernel/sys):
383 #
```

```
384 SYS_KMODS       +=

386 #
387 #       'User' Modules (/kernel/misc):
388 #
389 MISC_KMODS      += bootdev
390 MISC_KMODS      += dr_cpu
391 MISC_KMODS      += dr_io
392 MISC_KMODS      += dr_mem
393 MISC_KMODS      += ds
394 MISC_KMODS      += fault_iso
395 MISC_KMODS      += ldc
396 MISC_KMODS      += obpsym
397 MISC_KMODS      += platmod
398 MISC_KMODS      += platsvc
399 MISC_KMODS      += vis
400 MISC_KMODS      += pcie

402 #       md5 optimized for Niagara
403 #
404 MISC_KMODS      += md5

406 #
407 #       Brand modules
408 #
409 BRAND_KMODS     += sn1_brand s10_brand

411 #
412 #       Software Cryptographic Providers (/kernel/crypto):
413 #
414 CRYPTO_KMODS    += arcfour

416 #
417 #       generic-unix module (/kernel/genunix):
418 #
419 GENUNIX_KMODS   += genunix

426 #       'User' "Modules" excluded from the Full Kernel lint target:
427 #
428 $(CLOSED_BUILD)CLOSED_NLMISC_KMODS      += forthdebug

421 #
422 #       Modules eXcluded from the product:
423 #
424 XMODS           +=

426 #
427 #       cpu modules
428 #
429 CPU_KMODS       += generic niagara niagara2 vfalls kt

431 LINT_CPU_KMODS  += generic

433 #
434 #       Performance Counter BackEnd Modules (/usr/kernel/pcbe):
435 #
436 PCBE_KMODS      += niagara_pcbe
437 PCBE_KMODS      += niagara2_pcbe
438 PCBE_KMODS      += vfalls_pcbe
439 PCBE_KMODS      += kt_pcbe
```

```
**********************************************************
    2802 Wed Oct 16 17:41:14 2013
new/usr/src/uts/sun4v/sys/Makefile
4027 remove CLOSED_BUILD
4028 remove CLOSED_IS_PRESENT
4029 remove tonic build bits
Reviewed by: Andy Stormont <andyjstormont@gmail.com>
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#
# uts/sun4v/sys/Makefile
#
# include global definitions
UTSBASE = ../..

#
# include global definitions
#
include ../Makefile.sun4v

#
# Override defaults.
#
FILEMODE         = 644

SUN4_HDRS=                      \
        clock.h                 \
        cmp.h                   \
        cpc_ultra.h             \
        cpu_sgnblk_defs.h       \
        ddi_subrdefs.h          \
        dvma.h                  \
        eeprom.h                \
        fcode.h                 \
        idprom.h                \
        intr.h                  \
        intreg.h                \
        ivintr.h                \
        memlist_plat.h          \
        memnode.h               \
        nexusdebug.h            \
        prom_debug.h            \
        scb.h                   \
```



```
        sun4asi.h               \
        tod.h                   \
        trapstat.h              \
        vis.h                   \
        vm_machparam.h          \
        x_call.h                \
        xc_impl.h               \
        zsmach.h

CLOSED_SUN4_HDRS=                       \
        memtestio.h

HDRS=   \
        ds_pri.h                \
        ds_snmp.h               \
        hypervisor_api.h        \
        hsvc.h                  \
        machasi.h               \
        machclock.h             \
        machcpuvar.h            \
        mach_descrip.h          \
        machintreg.h            \
        machparam.h             \
        machsystm.h             \
        machthread.h            \
        mmu.h                   \
        niagaraasi.h            \
        niagararegs.h           \
        ntwdt.h                 \
        pte.h                   \
        prom_plat.h             \
        qcn.h                   \
        soft_state.h            \
        traptrace.h             \
        vlds.h

CLOSED_HDRS=    \
        memtestio_kt.h          \
        memtestio_ni.h          \
        memtestio_n2.h          \
        memtestio_v.h           \
        memtestio_vf.h

ROOTHDRS=               $(HDRS:%=$(USR_PSM_ISYS_DIR)/%)
$(CLOSED_BUILD)ROOTHDRS += $(CLOSED_HDRS:%=$(USR_PSM_ISYS_DIR)/%)

SUN4_ROOTHDRS=          $(SUN4_HDRS:%=$(USR_PSM_ISYS_DIR)/%)
$(CLOSED_BUILD)SUN4_ROOTHDRS += $(CLOSED_SUN4_HDRS:%=$(USR_PSM_ISYS_DIR)/%)

ROOTDIR=                $(ROOT)/usr/share/src
ROOTDIRS=               $(ROOTDIR)/uts $(ROOTDIR)/uts/$(PLATFORM)

ROOTLINK=               $(ROOTDIR)/uts/$(PLATFORM)/sys
LINKDEST=               ../../../../platform/$(PLATFORM)/include/sys

CHECKHDRS=              $(HDRS:%.h=%.check) \
                        $(SUN4_HDRS:%.h=%.cmncheck)

$(CLOSED_BUILD)CHECKHDRS +=                     \
        $(CLOSED_HDRS:%.h=%.check)              \
        $(CLOSED_SUN4_HDRS:%.h=%.cmncheck)

.KEEP_STATE:

.PARALLEL: $(CHECKHDRS) $(ROOTHDRS) $(SUN4_ROOTHDRS)
```

```
109 install_h: $(ROOTDIRS) .WAIT \
110                 $(ROOTHDRS) .WAIT \
111                 $(SUN4_ROOTHDRS) .WAIT $(ROOTLINK)

113 check:  $(CHECKHDRS)

115 #
116 # install rules
117 #
118 $(USR_PSM_ISYS_DIR)/%:  ../../sfmmu/sys/% $(USR_PSM_ISYS_DIR)
119         $(INS.file)

121 $(USR_PSM_ISYS_DIR)/%:  ../../sun4/sys/% $(USR_PSM_ISYS_DIR)
122         $(INS.file)

140 $(USR_PSM_ISYS_DIR)/%:  $(CLOSED)/uts/sun4/sys/% $(USR_PSM_ISYS_DIR)
141         $(INS.file)

143 $(USR_PSM_ISYS_DIR)/%:  $(CLOSED)/uts/sun4v/sys/% $(USR_PSM_ISYS_DIR)
144         $(INS.file)

124 $(ROOTDIRS):
125         $(INS.dir)

127 # -r because this used to be a directory and is now a link.
128 $(ROOTLINK):    $(ROOTDIRS)
129         -$(RM) -r $@; $(SYMLINK) $(LINKDEST) $@

131 mon/%.check:    mon/%.h
132         $(DOT_H_CHECK)

134 %.check:        ../../sfmmu/sys/%.h
135         $(DOT_H_CHECK)
158 %.check:        $(CLOSED)/uts/sun4v/sys/%.h
159         $(DOT_H_CHECK)
136 %.cmncheck:     ../../sun4/sys/%.h
137         $(DOT_H_CHECK)
162 %.cmncheck:     $(CLOSED)/uts/sun4/sys/%.h
163         $(DOT_H_CHECK)

139 FRC:

141 include ../../Makefile.targ
```