```
*************************************************************
   35097 Wed Oct  1 18:40:28 2014
new/usr/src/Makefile.master
5196 The cw wrapper restricts gcc to -O2
*************************************************************
#
# 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.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
# Copyright 2014 Gary Mills
#

#
# 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.
#
# __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=
RELEASE_BUILD=                          $(POUND_SIGN)
$(RELEASE_BUILD)NOT_RELEASE_BUILD=      $(POUND_SIGN)
PATCH_BUILD=                            $(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=)

# 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.
# __GNUC and GCC 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.
#

__SUNC=                 $(POUND_SIGN)
$(__SUNC)__GNUC=        $(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
PERL_VERSION=   5.10.0
PERL_PKGVERS=   -510
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
ASTBINDIR=      /usr/ast/bin
MSGCC=          $(ASTBINDIR)/msgcc

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 $<

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)

SAVEARGS=               -Wu,-save_args
amd64_STAND_FLAGS       += $(SAVEARGS)

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

#
# disable the incremental linker
ILDOFF=                 -xildoff
#
XDEPEND=                -xdepend
XFFLAG=                 -xF=%all
XESS=                   -xs
XSTRCONST=              -xstrconst

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

CERRWARN += -_gcc=-Wno-missing-braces
CERRWARN += -_gcc=-Wno-sign-compare
CERRWARN += -_gcc=-Wno-unknown-pragmas
CERRWARN += -_gcc=-Wno-unused-parameter
CERRWARN += -_gcc=-Wno-missing-field-initializers

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

# Suppress it: this warning generates many false alarms
CERRWARN +=     -_gcc=-Wno-uninitialized
```

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

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

# C99 mode
C99_ENABLE=     -xc99=%all
C99_DISABLE=    -xc99=%none
C99MODE=        $(C99_DISABLE)
C99LMODE=       $(C99MODE:-xc99%=-Xc99%)

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

sparc_ASFLAGS=  $(sparc_AS_XARCH)
sparcv9_ASFLAGS=$(sparcv9_AS_XARCH)
i386_ASFLAGS=   $(i386_AS_XARCH)
amd64_ASFLAGS=  $(amd64_AS_XARCH)

# Adjustments to specified optimization level
ADJUST_COPT=            -_gcc=-fno-strict-aliasing -_gcc=-fno-tree-vrp

#
sparc_COPTFLAG=         $(ADJUST_COPT) -xO3
sparcv9_COPTFLAG=       $(ADJUST_COPT) -xO3
i386_COPTFLAG=          $(ADJUST_COPT) -O
amd64_COPTFLAG=         $(ADJUST_COPT) -xO3
sparc_COPTFLAG=         -xO3
sparcv9_COPTFLAG=       -xO3
i386_COPTFLAG=          -O
amd64_COPTFLAG=         -xO3

COPTFLAG= $($(MACH)_COPTFLAG)
COPTFLAG64= $($(MACH64)_COPTFLAG)

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

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

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

#
# Default debug format for Sun Studio 11 is dwarf, so force it to
# generate stabs.
#
DEBUGFORMAT=    -xdebugformat=stabs
```

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

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

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

CTF_FLAGS_32    = $(CTF_FLAGS_$(MACH)) $(DEBUGFORMAT)
CTF_FLAGS_64    = $(CTF_FLAGS_$(MACH64)) $(DEBUGFORMAT)
CTF_FLAGS       = $(CTF_FLAGS_32)

#
# Flags used with genoffsets
#
GOFLAGS = -_noecho \
        $(CALLSYMS) \
        $(CDWARFSTR)

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

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

#
# tradeoff time for space (smaller is better)
#
sparc_SPACEFLAG         = -xspace -W0,-Lt
sparcv9_SPACEFLAG       = -xspace -W0,-Lt
i386_SPACEFLAG          = -xspace
amd64_SPACEFLAG         =

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

#
# 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   =
ELFWRAP_FLAGS64 =       -64

#
# Various mapfiles that are used throughout the build, and delivered to
# /usr/lib/ld.
#
MAPFILE.NED_i386 =      $(SRC)/common/mapfiles/common/map.noexdata
MAPFILE.NED_sparc =
MAPFILE.NED =           $(MAPFILE.NED_$(MACH))
MAPFILE.PGA =           $(SRC)/common/mapfiles/common/map.pagealign
MAPFILE.NES =           $(SRC)/common/mapfiles/common/map.noexstk
MAPFILE.FLT =           $(SRC)/common/mapfiles/common/map.filter
MAPFILE.LEX =           $(SRC)/common/mapfiles/common/map.lex.yy

#
# Generated mapfiles that are compiler specific, and used throughout the
# build.  These mapfiles are not delivered in /usr/lib/ld.
#
MAPFILE.NGB_sparc=      $(SRC)/common/mapfiles/gen/sparc_cc_map.noexeglobs
$(__GNUC64)MAPFILE.NGB_sparc= \
                        $(SRC)/common/mapfiles/gen/sparc_gcc_map.noexeglobs
MAPFILE.NGB_sparcv9=    $(SRC)/common/mapfiles/gen/sparcv9_cc_map.noexeglobs
$(__GNUC64)MAPFILE.NGB_sparcv9= \
                        $(SRC)/common/mapfiles/gen/sparcv9_gcc_map.noexeglobs
MAPFILE.NGB_i386=       $(SRC)/common/mapfiles/gen/i386_cc_map.noexeglobs
$(__GNUC64)MAPFILE.NGB_i386= \
                        $(SRC)/common/mapfiles/gen/i386_gcc_map.noexeglobs
MAPFILE.NGB_amd64=      $(SRC)/common/mapfiles/gen/amd64_cc_map.noexeglobs
$(__GNUC64)MAPFILE.NGB_amd64= \
                        $(SRC)/common/mapfiles/gen/amd64_gcc_map.noexeglobs
MAPFILE.NGB =           $(MAPFILE.NGB_$(MACH))

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

#
# LDLIBS32 and LDLIBS64 can be set in the environment to override the following
# assignments.
#
# These environment settings make sure that no libraries are searched outside
# of the local workspace proto area:
#       LDLIBS32=-YP,$ROOT/lib:$ROOT/usr/lib
#       LDLIBS64=-YP,$ROOT/lib/$MACH64:$ROOT/usr/lib/$MACH64
#
LDLIBS32 =      $(ENVLDLIBS1) $(ENVLDLIBS2) $(ENVLDLIBS3)
LDLIBS32 +=     $(ADJUNCT_PROTO:%=-L%/usr/lib -L%/lib)
LDLIBS.cmd =    $(LDLIBS32)
LDLIBS.lib =    $(LDLIBS32)

LDLIBS64 =      $(ENVLDLIBS1:%=%/$(MACH64)) \
                $(ENVLDLIBS2:%=%/$(MACH64)) \
                $(ENVLDLIBS3:%=%/$(MACH64))
LDLIBS64 +=     $(ADJUNCT_PROTO:%=-L%/usr/lib/$(MACH64) -L%/lib/$(MACH64))

#
# Define compilation macros.
#
COMPILE.c=      $(CC) $(CFLAGS) $(CPPFLAGS) -c
COMPILE64.c=    $(CC) $(CFLAGS64) $(CPPFLAGS) -c
COMPILE.cc=     $(CCC) $(CCFLAGS) $(CPPFLAGS) -c
COMPILE64.cc=   $(CCC) $(CCFLAGS64) $(CPPFLAGS) -c
COMPILE.s=      $(AS) $(ASFLAGS) $(AS_CPPFLAGS)
COMPILE64.s=    $(AS) $(ASFLAGS) $($(MACH64)_AS_XARCH) $(AS_CPPFLAGS)
COMPILE.d=      $(DTRACE) -G -32
COMPILE64.d=    $(DTRACE) -G -64
COMPILE.b=      $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))
COMPILE64.b=    $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))

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

#
# Link time macros
#
CCNEEDED                = -lC
CCEXTNEEDED             = -lCrun -lCstd
$(__GNUC)CCNEEDED       = -L$(GCCLIBDIR) -lstdc++ -lgcc_s
$(__GNUC)CCEXTNEEDED    = $(CCNEEDED)

LINK.c=         $(CC) $(CFLAGS) $(CPPFLAGS) $(LDFLAGS)
LINK64.c=       $(CC) $(CFLAGS64) $(CPPFLAGS) $(LDFLAGS)
NORUNPATH=      -norunpath -nolib
LINK.cc=        $(CCC) $(CCFLAGS) $(CPPFLAGS) $(NORUNPATH) \
                $(LDFLAGS) $(CCNEEDED)
LINK64.cc=      $(CCC) $(CCFLAGS64) $(CPPFLAGS) $(NORUNPATH) \
                $(LDFLAGS) $(CCNEEDED)
```

```
#
# lint macros
#
# Note that the undefine of __PRAGMA_REDEFINE_EXTNAME can be removed once
# ON is built with a version of lint that has the fix for 4484186.
#
ALWAYS_LINT_DEFS =      -errtags=yes -s
ALWAYS_LINT_DEFS +=     -erroff=E_PTRDIFF_OVERFLOW
ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_NARROW_CONV
ALWAYS_LINT_DEFS +=     -U__PRAGMA_REDEFINE_EXTNAME
ALWAYS_LINT_DEFS +=     $(C99LMODE)
ALWAYS_LINT_DEFS +=     -errsecurity=$(SECLEVEL)
ALWAYS_LINT_DEFS +=     -erroff=E_SEC_CREAT_WITHOUT_EXCL
ALWAYS_LINT_DEFS +=     -erroff=E_SEC_FORBIDDEN_WARN_CREAT
# XX64 -- really only needed for amd64 lint
ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_INT_TO_SMALL_INT
ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_CONST_TO_SMALL_INT
ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_TO_SMALL_INT
ALWAYS_LINT_DEFS +=     -erroff=E_CAST_TO_PTR_FROM_INT
ALWAYS_LINT_DEFS +=     -erroff=E_COMP_INT_WITH_LARGE_INT
ALWAYS_LINT_DEFS +=     -erroff=E_INTEGRAL_CONST_EXP_EXPECTED
ALWAYS_LINT_DEFS +=     -erroff=E_PASS_INT_TO_SMALL_INT
ALWAYS_LINT_DEFS +=     -erroff=E_PTR_CONV_LOSES_BITS

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

SECLEVEL=       core
LINT.c=         $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS) $(CPPFLAGS) \
                $(ALWAYS_LINT_DEFS)
LINT64.c=       $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS64) $(CPPFLAGS) \
                $(ALWAYS_LINT_DEFS)
LINT.s=         $(LINT.c)

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

# Define native compilation macros
#

# Base directory where compilers are loaded.
# Defined here so it can be overridden by developer.
#
SPRO_ROOT=              $(BUILD_TOOLS)/SUNWspro
SPRO_VROOT=             $(SPRO_ROOT)/SS12
GNU_ROOT=               $(SFW_ROOT)

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

# Specify platform compiler versions for languages
# that we use (currently only c and c++).
#
sparc_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
$(__GNUC)sparc_CC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
sparc_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
```

```
$(__GNUC)sparc_CCC=     $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
sparc_CPP=              /usr/ccs/lib/cpp
sparc_AS=               /usr/ccs/bin/as -xregsym=no
sparc_LD=               /usr/ccs/bin/ld
sparc_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

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

i386_CC=                $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
$(__GNUC)i386_CC=       $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
i386_CCC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
$(__GNUC)i386_CCC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
i386_CPP=               /usr/ccs/lib/cpp
i386_AS=                /usr/ccs/bin/as
$(__GNUC)i386_AS=       $(ONBLD_TOOLS)/bin/$(MACH)/aw
i386_LD=                /usr/ccs/bin/ld
i386_LINT=              $(SPRO_ROOT)/sunstudio12.1/bin/lint

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

NATIVECC=               $($(NATIVE_MACH)_CC)
NATIVECCC=              $($(NATIVE_MACH)_CCC)
NATIVECPP=              $($(NATIVE_MACH)_CPP)
NATIVEAS=               $($(NATIVE_MACH)_AS)
NATIVELD=               $($(NATIVE_MACH)_LD)
NATIVELINT=             $($(NATIVE_MACH)_LINT)

#
# Makefile.master.64 overrides these settings
#
CC=                     $(NATIVECC)
CCC=                    $(NATIVECCC)
CPP=                    $(NATIVECPP)
AS=                     $(NATIVEAS)
LD=                     $(NATIVELD)
LINT=                   $(NATIVELINT)

# The real compilers used for this build
CW_CC_CMD=              $(CC) -_compiler
CW_CCC_CMD=             $(CCC) -_compiler
REAL_CC=                $(CW_CC_CMD:sh)
REAL_CCC=               $(CW_CCC_CMD:sh)

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

BDIRECT=        -Bdirect
BDYNAMIC=       -Bdynamic
BLOCAL=         -Blocal
BNODIRECT=      -Bnodirect
BREDUCE=        -Breduce
BSTATIC=        -Bstatic
```

```
ZDEFS=          -zdefs
ZDIRECT=        -zdirect
ZIGNORE=        -zignore
ZINITFIRST=     -zinitfirst
ZINTERPOSE=     -zinterpose
ZLAZYLOAD=      -zlazyload
ZLOADFLTR=      -zloadfltr
ZMULDEFS=       -zmuldefs
ZNODEFAULTLIB=  -znodefaultlib
ZNODEFS=        -znodefs
ZNODELETE=      -znodelete
ZNODLOPEN=      -znodlopen
ZNODUMP=        -znodump
ZNOLAZYLOAD=    -znolazyload
ZNOLDYNSYM=     -znoldynsym
ZNORELOC=       -znoreloc
ZNOVERSION=     -znoversion
ZRECORD=        -zrecord
ZREDLOCSYM=     -zredlocsym
ZTEXT=          -ztext
ZVERBOSE=       -zverbose

GSHARED=        -G
CCMT=           -mt

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

sparc_C_PICFLAGS =      -K pic
sparcv9_C_PICFLAGS =    -K pic
i386_C_PICFLAGS =       -K pic
amd64_C_PICFLAGS =      -K pic
C_PICFLAGS =            $($(MACH)_C_PICFLAGS)
C_PICFLAGS64 =          $($(MACH64)_C_PICFLAGS)

sparc_C_BIGPICFLAGS =   -K PIC
sparcv9_C_BIGPICFLAGS = -K PIC
i386_C_BIGPICFLAGS =    -K PIC
amd64_C_BIGPICFLAGS =   -K PIC
C_BIGPICFLAGS =         $($(MACH)_C_BIGPICFLAGS)
C_BIGPICFLAGS64 =       $($(MACH64)_C_BIGPICFLAGS)

# CC requires there to be no space between '-K' and 'pic' or 'PIC'.
sparc_CC_PICFLAGS =     -Kpic
sparcv9_CC_PICFLAGS =   -KPIC
i386_CC_PICFLAGS =      -Kpic
amd64_CC_PICFLAGS =     -Kpic
CC_PICFLAGS =           $($(MACH)_CC_PICFLAGS)
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} -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

.h.check:
	$(DOT_H_CHECK)

.x.check:
	$(DOT_X_CHECK)

.xml.xmlchk:
	$(MANIFEST_CHECK)

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

```
**********************************************************
   45264 Wed Oct  1 18:40:29 2014
new/usr/src/tools/cw/cw.c
5196 The cw wrapper restricts gcc to -O2
**********************************************************
/*
 * 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 2014 Gary Mills
 *
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*
 * Wrapper for the GNU C compiler to make it accept the Sun C compiler
 * arguments where possible.
 *
 * Since the translation is inexact, this is something of a work-in-progress.
 *
 */

/* If you modify this file, you must increment CW_VERSION */
#define CW_VERSION      "1.30"
#define CW_VERSION      "1.29"

/*
 * -#           Verbose mode
 * -###         Show compiler commands built by driver, no compilation
 * -A<name[(tokens)]>   Preprocessor predicate assertion
 * -B<[static|dynamic]> Specify dynamic or static binding
 * -C           Prevent preprocessor from removing comments
 * -c           Compile only - produce .o files, suppress linking
 * -cg92        Alias for -xtarget=ss1000
 * -D<name[=token]>     Associate name with token as if by #define
 * -d[y|n]      dynamic [-dy] or static [-dn] option to linker
 * -E           Compile source through preprocessor only, output to stdout
 * -erroff=<t>  Suppress warnings specified by tags t(%none, %all, <tag list>)
 * -errtags=<a> Display messages with tags a(no, yes)
 * -errwarn=<t> Treats warnings specified by tags t(%none, %all, <tag list>)
 *              as errors
 * -fast        Optimize using a selection of options
 * -fd          Report old-style function definitions and declarations
 * -features=zla        Allow zero-length arrays
 * -flags       Show this summary of compiler options
 * -fnonstd     Initialize floating-point hardware to non-standard preferences
 * -fns[=<yes|no>] Select non-standard floating point mode
```

```
 * -fprecision=<p> Set FP rounding precision mode p(single, double, extended)
 * -fround=<r>  Select the IEEE rounding mode in effect at startup
 * -fsimple[=<n>] Select floating-point optimization preferences <n>
 * -fsingle     Use single-precision arithmetic (-Xt and -Xs modes only)
 * -ftrap=<t>   Select floating-point trapping mode in effect at startup
 * -fstore      force floating pt. values to target precision on assignment
 * -G           Build a dynamic shared library
 * -g           Compile for debugging
 * -H           Print path name of each file included during compilation
 * -h <name>    Assign <name> to generated dynamic shared library
 * -I<dir>      Add <dir> to preprocessor #include file search path
 * -i           Passed to linker to ignore any LD_LIBRARY_PATH setting
 * -keeptmp     Keep temporary files created during compilation
 * -KPIC        Compile position independent code with 32-bit addresses
 * -Kpic        Compile position independent code
 * -L<dir>      Pass to linker to add <dir> to the library search path
 * -l<name>     Link with library lib<name>.a or lib<name>.so
 * -mc          Remove duplicate strings from .comment section of output files
 * -mr          Remove all strings from .comment section of output files
 * -mr,"string" Remove all strings and append "string" to .comment section
 * -mt          Specify options needed when compiling multi-threaded code
 * -native      Find available processor, generate code accordingly
 * -nofstore    Do not force floating pt. values to target precision
 *              on assignment
 * -nolib       Same as -xnolib
 * -noqueue     Disable queuing of compiler license requests
 * -norunpath   Do not build in a runtime path for shared libraries
 * -O           Use default optimization level (-xO2 or -xO3. Check man page.)
 * -o <outputfile> Set name of output file to <outputfile>
 * -P           Compile source through preprocessor only, output to .i  file
 * -PIC         Alias for -KPIC or -xcode=pic32
 * -p           Compile for profiling with prof
 * -pic         Alias for -Kpic or -xcode=pic13
 * -Q[y|n]      Emit/don't emit identification info to output file
 * -qp          Compile for profiling with prof
 * -R<dir[:dir]> Build runtime search path list into executable
 * -S           Compile and only generate assembly code (.s)
 * -s           Strip symbol table from the executable file
 * -t           Turn off duplicate symbol warnings when linking
 * -U<name>     Delete initial definition of preprocessor symbol <name>
 * -V           Report version number of each compilation phase
 * -v           Do stricter semantic checking
 * -W<c>,<arg>  Pass <arg> to specified component <c> (a,l,m,p,0,2,h,i,u)
 * -w           Suppress compiler warning messages
 * -Xa          Compile assuming ANSI C conformance, allow K & R extensions
 *              (default mode)
 * -Xc          Compile assuming strict ANSI C conformance
 * -Xs          Compile assuming (pre-ANSI) K & R C style code
 * -Xt          Compile assuming K & R conformance, allow ANSI C
 * -x386        Generate code for the 80386 processor
 * -x486        Generate code for the 80486 processor
 * -xarch=<a>   Specify target architecture instruction set
 * -xbuiltin[=<b>] When profitable inline, or substitute intrinisic functions
 *              for system functions, b={%all,%none}
 * -xCC         Accept C++ style comments
 * -xchar_byte_order=<o> Specify multi-char byte order <o> (default, high, low)
 * -xchip=<c>   Specify the target processor for use by the optimizer
 * -xcode=<c>   Generate different code for forming addresses
 * -xcrossfile[=<n>] Enable optimization and inlining across source files,
 *              n={0|1}
 * -xe          Perform only syntax/semantic checking, no code generation
 * -xF          Compile for later mapfile reordering or unused section
 *              elimination
 * -xhelp=<f>   Display on-line help information f(flags, readme, errors)
 * -xildoff     Cancel -xildon
 * -xildon      Enable use of the incremental linker, ild
 * -xinline=[<a>,...,<a>]  Attempt inlining of specified user routines,
 *              <a>={%auto,func,no%func}
 * -xlibmieee   Force IEEE 754 return values for math routines in
 *              exceptional cases
 * -xlibmil     Inline selected libm math routines for optimization
 * -xlic_lib=sunperf    Link in the Sun supplied performance libraries
 * -xlicinfo    Show license server information
 * -xM          Generate makefile dependencies
 * -xM1         Generate makefile dependencies, but exclude /usr/include
 * -xmaxopt=[off,1,2,3,4,5] maximum optimization level allowed on #pragma opt
 * -xnolib      Do not link with default system libraries
 * -xnolibmil   Cancel -xlibmil on command line
 * -xO<n>       Generate optimized code (n={1|2|3|4|5})
 * -xP          Print prototypes for function definitions
 * -xpentium    Generate code for the pentium processor
 * -xpg         Compile for profiling with gprof
 * -xprofile=<p> Collect data for a profile or use a profile to optimize
 *              <p>={{collect,use}[:<path>],tcov}
 * -xregs=<r>   Control register allocation
 * -xs          Allow debugging without object (.o) files
 * -xsb         Compile for use with the WorkShop source browser
 * -xsbfast     Generate only WorkShop source browser info, no compilation
 * -xsfpconst   Represent unsuffixed floating point constants as single
 *              precision
 * -xspace      Do not do optimizations that increase code size
 * -xstrconst   Place string literals into read-only data segment
 * -xtarget=<t> Specify target system for optimization
 * -xtemp=<dir> Set directory for temporary files to <dir>
 * -xtime       Report the execution time for each compilation phase
 * -xtransition Emit warnings for differences between K&R C and ANSI C
 * -xtrigraphs[=<yes|no>] Enable|disable trigraph translation
 * -xunroll=n   Enable unrolling loops n times where possible
 * -Y<c>,<dir>  Specify <dir> for location of component <c> (a,l,m,p,0,h,i,u)
 * -YA,<dir>    Change default directory searched for components
 * -YI,<dir>    Change default directory searched for include files
 * -YP,<dir>    Change default directory for finding libraries files
 * -YS,<dir>    Change default directory for startup object files
 */

/*
 * Translation table:
 */
/*
 * -#                           -v
 * -###                         error
 * -A<name[(tokens)]>           pass-thru
 * -B<[static|dynamic]>         pass-thru (syntax error for anything else)
 * -C                           pass-thru
 * -c                           pass-thru
 * -cg92                        -m32 -mcpu=v8 -mtune=supersparc (SPARC only)
 * -D<name[=token]>             pass-thru
 * -dy or -dn                   -Wl,-dy or -Wl,-dn
 * -E                           pass-thru
 * -erroff=E_EMPTY_TRANSLATION_UNIT ignore
 * -errtags=%all                -Wall
 * -errwarn=%all                -Werror else -Wno-error
 * -fast                        error
 * -fd                          error
 * -features=zla                ignore
 * -flags                       --help
 * -fnonstd                     error
 * -fns[=<yes|no>]              error
 * -fprecision=<p>              error
 * -fround=<r>                  error
 * -fsimple[=<n>]               error
 * -fsingle[=<n>]               error
```

```
 * -ftrap=<t>                   error
 * -fstore                      error
 * -G                           pass-thru
 * -g                           pass-thru
 * -H                           pass-thru
 * -h <name>                    pass-thru
 * -I<dir>                      pass-thru
 * -i                           pass-thru
 * -keeptmp                     -save-temps
 * -KPIC                        -fPIC
 * -Kpic                        -fpic
 * -L<dir>                      pass-thru
 * -l<name>                     pass-thru
 * -mc                          error
 * -mr                          error
 * -mr,"string"                 error
 * -mt                          -D_REENTRANT
 * -native                      error
 * -nofstore                    error
 * -nolib                       -nodefaultlibs
 * -noqueue                     ignore
 * -norunpath                   ignore
 * -O                           -O1 (Check the man page to be certain)
 * -o <outputfile>              pass-thru
 * -P                           -E -o filename.i (or error)
 * -PIC                         -fPIC (C++ only)
 * -p                           pass-thru
 * -pic                         -fpic (C++ only)
 * -Q[y|n]                      error
 * -qp                          -p
 * -R<dir[:dir]>                pass-thru
 * -S                           pass-thru
 * -s                           -Wl,-s
 * -t                           -Wl,-t
 * -U<name>                     pass-thru
 * -V                           --version
 * -v                           -Wall
 * -Wa,<arg>                    pass-thru
 * -Wp,<arg>                    pass-thru except -xc99=<a>
 * -Wl,<arg>                    pass-thru
 * -W{m,0,2,h,i,u>              error/ignore
 * -Wu,-xmodel=kernel           -ffreestanding -mcmodel=kernel -mno-red-zone
 * -xmodel=kernel               -ffreestanding -mcmodel=kernel -mno-red-zone
 * -Wu,-save_args               -msave-args
 * -w                           pass-thru
 * -Xa                          -std=iso9899:199409 or -ansi
 * -Xc                          -ansi -pedantic
 * -Xt                          error
 * -Xs                          -traditional -std=c89
 * -x386                        -march=i386 (x86 only)
 * -x486                        -march=i486 (x86 only)
 * -xarch=<a>                   table
 * -xbuiltin[=<b>]              -fbuiltin (-fno-builtin otherwise)
 * -xCC                         ignore
 * -xchar_byte_order=<o>        error
 * -xchip=<c>                   table
 * -xcode=<c>                   table
 * -xdebugformat=<format>       ignore (always use dwarf-2 for gcc)
 * -xcrossfile[=<n>]            ignore
 * -xe                          error
 * -xF                          error
 * -xhelp=<f>                   error
 * -xildoff                     ignore
 * -xildon                      ignore
 * -xinline                     ignore
 * -xlibmieee                   error
 * -xlibmil                     error
 * -xlic_lib=sunperf            error
 * -xM                          -M
 * -xM1                         -MM
 * -xmaxopt=[...]               error
 * -xnolib                      -nodefaultlibs
 * -xnolibmil                   error
 * -xO<n>                       -O<n>
 * -xP                          error
 * -xpentium                    -march=pentium (x86 only)
 * -xpg                         error
 * -xprofile=<p>                error
 * -xregs=<r>                   table
 * -xs                          error
 * -xsb                         error
 * -xsbfast                     error
 * -xsfpconst                   error
 * -xspace                      ignore (-not -Os)
 * -xstrconst                   ignore
 * -xtarget=<t>                 table
 * -xtemp=<dir>                 error
 * -xtime                       error
 * -xtransition                 -Wtransition
 * -xtrigraphs=<yes|no>         -trigraphs -notrigraphs
 * -xunroll=n                   error
 * -W0,-xdbggen=no%usedonly     -fno-eliminate-unused-debug-symbols
 *                              -fno-eliminate-unused-debug-types
 * -Y<c>,<dir>                  error
 * -YA,<dir>                    error
 * -YI,<dir>                    -nostdinc -I<dir>
 * -YP,<dir>                    error
 * -YS,<dir>                    error
 */

#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#include <fcntl.h>
#include <errno.h>
#include <stdarg.h>
#include <sys/utsname.h>
#include <sys/param.h>
#include <sys/isa_defs.h>
#include <sys/wait.h>
#include <sys/stat.h>

#define CW_F_CXX        0x01
#define CW_F_SHADOW     0x02
#define CW_F_EXEC       0x04
#define CW_F_ECHO       0x08
#define CW_F_XLATE      0x10
#define CW_F_PROG       0x20

typedef enum cw_compiler {
        CW_C_CC = 0,
        CW_C_GCC
} cw_compiler_t;
```

______***unchanged_portion_omitted_***

```
static void
optim_disable(struct aelist *h, int level)
{
        if (level >= 2) {
```

```
                newae(h, "-fno-strict-aliasing");
                newae(h, "-fno-unit-at-a-time");
                newae(h, "-fno-optimize-sibling-calls");
        }
}

/* ARGSUSED */
static void
Xamode(struct aelist *h)
{
}
```

______***unchanged_portion_omitted_***

```
static void
do_gcc(cw_ictx_t *ctx)
{
        int c;
        int pic = 0, nolibc = 0;
        int in_output = 0, seen_o = 0, c_files = 0;
        cw_op_t op = CW_O_LINK;
        char *model = NULL;
        int     mflag = 0;

        if (ctx->i_flags & CW_F_PROG) {
                newae(ctx->i_ae, "--version");
                return;
        }

        newae(ctx->i_ae, "-fident");
        newae(ctx->i_ae, "-finline");
        newae(ctx->i_ae, "-fno-inline-functions");
        newae(ctx->i_ae, "-fno-builtin");
        newae(ctx->i_ae, "-fno-asm");
        newae(ctx->i_ae, "-fdiagnostics-show-option");
        newae(ctx->i_ae, "-nodefaultlibs");

#if defined(__sparc)
        /*
         * The SPARC ldd and std instructions require 8-byte alignment of
         * their address operand.  gcc correctly uses them only when the
         * ABI requires 8-byte alignment; unfortunately we have a number of
         * pieces of buggy code that doesn't conform to the ABI.  This
         * flag makes gcc work more like Studio with -xmemalign=4.
         */
        newae(ctx->i_ae, "-mno-integer-ldd-std");
#endif

        /*
         * This is needed because 'u' is defined
         * under a conditional on 'sun'.  Should
         * probably just remove the conditional,
         * or make it be dependent on '__sun'.
         *
         * -Dunix is also missing in enhanced ANSI mode
         */
        newae(ctx->i_ae, "-D__sun");

        /*
         * Walk the argument list, translating as we go ..
         */

        while (--ctx->i_oldargc > 0) {
                char *arg = *++ctx->i_oldargv;
                size_t arglen = strlen(arg);

                if (*arg == '-') {
                        arglen--;
                } else {
                        /*
                         * Discard inline files that gcc doesn't grok
                         */
                        if (!in_output && arglen > 3 &&
                            strcmp(arg + arglen - 3, ".il") == 0)
                                continue;

                        if (!in_output && arglen > 2 &&
                            arg[arglen - 2] == '.' &&
                            (arg[arglen - 1] == 'S' || arg[arglen - 1] == 's' ||
                            arg[arglen - 1] == 'c' || arg[arglen - 1] == 'i'))
                                c_files++;

                        /*
                         * Otherwise, filenames and partial arguments
                         * are passed through for gcc to chew on.  However,
                         * output is always discarded for the secondary
                         * compiler.
                         */
                        if ((ctx->i_flags & CW_F_SHADOW) && in_output)
                                newae(ctx->i_ae, ctx->i_discard);
                        else
                                newae(ctx->i_ae, arg);
                        in_output = 0;
                        continue;
                }

                if (ctx->i_flags & CW_F_CXX) {
                        if (strncmp(arg, "-compat=", 8) == 0) {
                                /* discard -compat=4 and -compat=5 */
                                continue;
                        }
                        if (strcmp(arg, "-Qoption") == 0) {
                                /* discard -Qoption and its two arguments */
                                if (ctx->i_oldargc < 3)
                                        error(arg);
                                ctx->i_oldargc -= 2;
                                ctx->i_oldargv += 2;
                                continue;
                        }
                        if (strcmp(arg, "-xwe") == 0) {
                                /* turn warnings into errors */
                                newae(ctx->i_ae, "-Werror");
                                continue;
                        }
                        if (strcmp(arg, "-noex") == 0) {
                                /* no exceptions */
                                newae(ctx->i_ae, "-fno-exceptions");
                                /* no run time type descriptor information */
                                newae(ctx->i_ae, "-fno-rtti");
                                continue;
                        }
                        if (strcmp(arg, "-pic") == 0) {
                                newae(ctx->i_ae, "-fpic");
                                pic = 1;
                                continue;
                        }
                        if (strcmp(arg, "-PIC") == 0) {
                                newae(ctx->i_ae, "-fPIC");
                                pic = 1;
                                continue;
                        }
                        if (strcmp(arg, "-norunpath") == 0) {
                                /* gcc has no corresponding option */
```

```
                                continue;
                        }
                        if (strcmp(arg, "-nolib") == 0) {
                                /* -nodefaultlibs is on by default */
                                nolibc = 1;
                                continue;
                        }
#if defined(__sparc)
                        if (strcmp(arg, "-cg92") == 0) {
                                mflag |= xlate_xtb(ctx->i_ae, "v8");
                                xlate(ctx->i_ae, "super", xchip_tbl);
                                continue;
                        }
#endif  /* __sparc */
                }

                switch ((c = arg[1])) {
                case '_':
                        if (strcmp(arg, "-_noecho") == 0)
                                ctx->i_flags &= ~CW_F_ECHO;
                        else if (strncmp(arg, "-_cc=", 5) == 0 ||
                            strncmp(arg, "-_CC=", 5) == 0)
                                /* EMPTY */;
                        else if (strncmp(arg, "-_gcc=", 6) == 0 ||
                            strncmp(arg, "-_g++=", 6) == 0)
                                newae(ctx->i_ae, arg + 6);
                        else
                                error(arg);
                        break;
                case '#':
                        if (arglen == 1) {
                                newae(ctx->i_ae, "-v");
                                break;
                        }
                        error(arg);
                        break;
                case 'g':
                        newae(ctx->i_ae, "-gdwarf-2");
                        break;
                case 'E':
                        if (arglen == 1) {
                                newae(ctx->i_ae, "-xc");
                                newae(ctx->i_ae, arg);
                                op = CW_O_PREPROCESS;
                                nolibc = 1;
                                break;
                        }
                        error(arg);
                        break;
                case 'c':
                case 'S':
                        if (arglen == 1) {
                                op = CW_O_COMPILE;
                                nolibc = 1;
                        }
                        /* FALLTHROUGH */
                case 'C':
                case 'H':
                case 'p':
                        if (arglen == 1) {
                                newae(ctx->i_ae, arg);
                                break;
                        }
                        error(arg);
                        break;
                case 'A':
```

```
		case 'h':
		case 'I':
		case 'i':
		case 'L':
		case 'l':
		case 'R':
		case 'U':
		case 'u':
		case 'w':
			newae(ctx->i_ae, arg);
			break;
		case 'o':
			seen_o = 1;
			if (arglen == 1) {
				in_output = 1;
				newae(ctx->i_ae, arg);
			} else if (ctx->i_flags & CW_F_SHADOW) {
				newae(ctx->i_ae, "-o");
				newae(ctx->i_ae, ctx->i_discard);
			} else {
				newae(ctx->i_ae, arg);
			}
			break;
		case 'D':
			newae(ctx->i_ae, arg);
			/*
			 * XXX  Clearly a hack ... do we need _KADB too?
			 */
			if (strcmp(arg, "-D_KERNEL") == 0 ||
			    strcmp(arg, "-D_BOOT") == 0)
				newae(ctx->i_ae, "-ffreestanding");
			break;
		case 'd':
			if (arglen == 2) {
				if (strcmp(arg, "-dy") == 0) {
					newae(ctx->i_ae, "-Wl,-dy");
					break;
				}
				if (strcmp(arg, "-dn") == 0) {
					newae(ctx->i_ae, "-Wl,-dn");
					break;
				}
			}
			if (strcmp(arg, "-dalign") == 0) {
				/*
				 * -dalign forces alignment in some cases;
				 * gcc does not need any flag to do this.
				 */
				break;
			}
			error(arg);
			break;
		case 'e':
			if (strcmp(arg,
			    "-erroff=E_EMPTY_TRANSLATION_UNIT") == 0) {
				/*
				 * Accept but ignore this -- gcc doesn't
				 * seem to complain about empty translation
				 * units
				 */
				break;
			}
			/* XX64 -- ignore all -erroff= options, for now */
			if (strncmp(arg, "-erroff=", 8) == 0)
				break;
			if (strcmp(arg, "-errtags=yes") == 0) {
```

```
				warnings(ctx->i_ae);
				break;
			}
			if (strcmp(arg, "-errwarn=%all") == 0) {
				newae(ctx->i_ae, "-Werror");
				break;
			}
			error(arg);
			break;
		case 'f':
			if (strcmp(arg, "-flags") == 0) {
				newae(ctx->i_ae, "--help");
				break;
			}
			if (strncmp(arg, "-features=zla", 13) == 0) {
				/*
				 * Accept but ignore this -- gcc allows
				 * zero length arrays.
				 */
				break;
			}
			error(arg);
			break;
		case 'G':
			newae(ctx->i_ae, "-shared");
			nolibc = 1;
			break;
		case 'k':
			if (strcmp(arg, "-keeptmp") == 0) {
				newae(ctx->i_ae, "-save-temps");
				break;
			}
			error(arg);
			break;
		case 'K':
			if (arglen == 1) {
				if ((arg = *++ctx->i_oldargv) == NULL ||
				    *arg == '\0')
					error("-K");
				ctx->i_oldargc--;
			} else {
				arg += 2;
			}
			if (strcmp(arg, "pic") == 0) {
				newae(ctx->i_ae, "-fpic");
				pic = 1;
				break;
			}
			if (strcmp(arg, "PIC") == 0) {
				newae(ctx->i_ae, "-fPIC");
				pic = 1;
				break;
			}
			error("-K");
			break;
		case 'm':
			if (strcmp(arg, "-mt") == 0) {
				newae(ctx->i_ae, "-D_REENTRANT");
				break;
			}
			if (strcmp(arg, "-m64") == 0) {
				newae(ctx->i_ae, "-m64");
#if defined(__x86)
				newae(ctx->i_ae, "-mtune=opteron");
#endif
				mflag |= M64;
				break;
			}
			if (strcmp(arg, "-m32") == 0) {
				newae(ctx->i_ae, "-m32");
				mflag |= M32;
				break;
			}
			error(arg);
			break;
		case 'B':	/* linker options */
		case 'M':
		case 'z':
			{
				char *opt;
				size_t len;
				char *s;

				if (arglen == 1) {
					opt = *++ctx->i_oldargv;
					if (opt == NULL || *opt == '\0')
						error(arg);
					ctx->i_oldargc--;
				} else {
					opt = arg + 2;
				}
				len = strlen(opt) + 7;
				if ((s = malloc(len)) == NULL)
					nomem();
				(void) snprintf(s, len, "-Wl,-%c%s", c, opt);
				newae(ctx->i_ae, s);
				free(s);
			}
			break;
		case 'n':
			if (strcmp(arg, "-noqueue") == 0) {
				/*
				 * Horrid license server stuff - n/a
				 */
				break;
			}
			error(arg);
			break;
		case 'O':
			if (arglen == 1) {
				newae(ctx->i_ae, "-O");
				break;
			}
			error(arg);
			break;
		case 'P':
			/*
			 * We could do '-E -o filename.i', but that's hard,
			 * and we don't need it for the case that's triggering
			 * this addition.  We'll require the user to specify
			 * -o in the Makefile.  If they don't they'll find out
			 * in a hurry.
			 */
			newae(ctx->i_ae, "-E");
			op = CW_O_PREPROCESS;
			nolibc = 1;
			break;
		case 'q':
			if (strcmp(arg, "-qp") == 0) {
				newae(ctx->i_ae, "-p");
				break;
			}
```

```
			error(arg);
			break;
		case 's':
			if (arglen == 1) {
				newae(ctx->i_ae, "-Wl,-s");
				break;
			}
			error(arg);
			break;
		case 't':
			if (arglen == 1) {
				newae(ctx->i_ae, "-Wl,-t");
				break;
			}
			error(arg);
			break;
		case 'V':
			if (arglen == 1) {
				ctx->i_flags &= ~CW_F_ECHO;
				newae(ctx->i_ae, "--version");
				break;
			}
			error(arg);
			break;
		case 'v':
			if (arglen == 1) {
				warnings(ctx->i_ae);
				break;
			}
			error(arg);
			break;
		case 'W':
			if (strncmp(arg, "-Wp,-xc99", 9) == 0) {
				/*
				 * gcc's preprocessor will accept c99
				 * regardless, so accept and ignore.
				 */
				break;
			}
			if (strncmp(arg, "-Wa,", 4) == 0 ||
			    strncmp(arg, "-Wp,", 4) == 0 ||
			    strncmp(arg, "-Wl,", 4) == 0) {
				newae(ctx->i_ae, arg);
				break;
			}
			if (strcmp(arg, "-W0,-xc99=pragma") == 0) {
				/* (undocumented) enables _Pragma */
				break;
			}
			if (strcmp(arg, "-W0,-xc99=%none") == 0) {
				/*
				 * This is a polite way of saying
				 * "no c99 constructs allowed!"
				 * For now, just accept and ignore this.
				 */
				break;
			}
			if (strcmp(arg, "-W0,-noglobal") == 0 ||
			    strcmp(arg, "-W0,-xglobalstatic") == 0) {
				/*
				 * gcc doesn't prefix local symbols
				 * in debug mode, so this is not needed.
				 */
				break;
			}
			if (strcmp(arg, "-W0,-Lt") == 0) {
```

```
                                /*
                                 * Generate tests at the top of loops.
                                 * There is no direct gcc equivalent, ignore.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-W0,-xdbggen=no%usedonly") == 0) {
                                newae(ctx->i_ae,
                                    "-fno-eliminate-unused-debug-symbols");
                                newae(ctx->i_ae,
                                    "-fno-eliminate-unused-debug-types");
                                break;
                        }
                        if (strcmp(arg, "-W2,-xwrap_int") == 0) {
                                /*
                                 * Use the legacy behaviour (pre-SS11)
                                 * for integer wrapping.
                                 * gcc does not need this.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-W2,-Rcond_elim") == 0) {
                                /*
                                 * Elimination and expansion of conditionals;
                                 * gcc has no direct equivalent.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-Wd,-xsafe=unboundsym") == 0) {
                                /*
                                 * Prevents optimizing away checks for
                                 * unbound weak symbol addresses.  gcc does
                                 * not do this, so it's not needed.
                                 */
                                break;
                        }
                        if (strncmp(arg, "-Wc,-xcode=", 11) == 0) {
                                xlate(ctx->i_ae, arg + 11, xcode_tbl);
                                if (strncmp(arg + 11, "pic", 3) == 0)
                                        pic = 1;
                                break;
                        }
                        if (strncmp(arg, "-Wc,-Qiselect", 13) == 0) {
                                /*
                                 * Prevents insertion of register symbols.
                                 * gcc doesn't do this, so ignore it.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-Wc,-Qassembler-ounrefsym=0") == 0) {
                                /*
                                 * Prevents optimizing away of static variables.
                                 * gcc does not do this, so it's not needed.
                                 */
                                break;
                        }
#if defined(__x86)
                        if (strcmp(arg, "-Wu,-xmodel=kernel") == 0) {
                                newae(ctx->i_ae, "-ffreestanding");
                                newae(ctx->i_ae, "-mno-red-zone");
                                model = "-mcmodel=kernel";
                                nolibc = 1;
                                break;
                        }
                        if (strcmp(arg, "-Wu,-save_args") == 0) {
                                newae(ctx->i_ae, "-msave-args");
```

```
                                break;
                        }
#endif  /* __x86 */
                        error(arg);
                        break;
                case 'X':
                        if (strcmp(arg, "-Xa") == 0 ||
                            strcmp(arg, "-Xt") == 0) {
                                Xamode(ctx->i_ae);
                                break;
                        }
                        if (strcmp(arg, "-Xc") == 0) {
                                Xcmode(ctx->i_ae);
                                break;
                        }
                        if (strcmp(arg, "-Xs") == 0) {
                                Xsmode(ctx->i_ae);
                                break;
                        }
                        error(arg);
                        break;
                case 'x':
                        if (arglen == 1)
                                error(arg);
                        switch (arg[2]) {
#if defined(__x86)
                        case '3':
                                if (strcmp(arg, "-x386") == 0) {
                                        newae(ctx->i_ae, "-march=i386");
                                        break;
                                }
                                error(arg);
                                break;
                        case '4':
                                if (strcmp(arg, "-x486") == 0) {
                                        newae(ctx->i_ae, "-march=i486");
                                        break;
                                }
                                error(arg);
                                break;
#endif  /* __x86 */
                        case 'a':
                                if (strncmp(arg, "-xarch=", 7) == 0) {
                                        mflag |= xlate_xtb(ctx->i_ae, arg + 7);
                                        break;
                                }
                                error(arg);
                                break;
                        case 'b':
                                if (strncmp(arg, "-xbuiltin=", 10) == 0) {
                                        if (strcmp(arg + 10, "%all"))
                                                newae(ctx->i_ae, "-fbuiltin");
                                        break;
                                }
                                error(arg);
                                break;
                        case 'C':
                                /* Accept C++ style comments -- ignore */
                                if (strcmp(arg, "-xCC") == 0)
                                        break;
                                error(arg);
                                break;
                        case 'c':
                                if (strncmp(arg, "-xc99=%all", 10) == 0) {
                                        newae(ctx->i_ae, "-std=gnu99");
                                        break;
                                }
                                if (strncmp(arg, "-xc99=%none", 11) == 0) {
                                        newae(ctx->i_ae, "-std=gnu89");
                                        break;
                                }
                                if (strncmp(arg, "-xchip=", 7) == 0) {
                                        xlate(ctx->i_ae, arg + 7, xchip_tbl);
                                        break;
                                }
                                if (strncmp(arg, "-xcode=", 7) == 0) {
                                        xlate(ctx->i_ae, arg + 7, xcode_tbl);
                                        if (strncmp(arg + 7, "pic", 3) == 0)
                                                pic = 1;
                                        break;
                                }
                                if (strncmp(arg, "-xcache=", 8) == 0)
                                        break;
                                if (strncmp(arg, "-xcrossfile", 11) == 0)
                                        break;
                                error(arg);
                                break;
                        case 'd':
                                if (strcmp(arg, "-xdepend") == 0)
                                        break;
                                if (strncmp(arg, "-xdebugformat=", 14) == 0)
                                        break;
                                error(arg);
                                break;
                        case 'F':
                                /*
                                 * Compile for mapfile reordering, or unused
                                 * section elimination, syntax can be -xF or
                                 * more complex, like -xF=%all -- ignore.
                                 */
                                if (strncmp(arg, "-xF", 3) == 0)
                                        break;
                                error(arg);
                                break;
                        case 'i':
                                if (strncmp(arg, "-xinline", 8) == 0)
                                        /* No inlining; ignore */
                                        break;
                                if (strcmp(arg, "-xildon") == 0 ||
                                    strcmp(arg, "-xildoff") == 0)
                                        /* No incremental linking; ignore */
                                        break;
                                error(arg);
                                break;
#if defined(__x86)
                        case 'm':
                                if (strcmp(arg, "-xmodel=kernel") == 0) {
                                        newae(ctx->i_ae, "-ffreestanding");
                                        newae(ctx->i_ae, "-mno-red-zone");
                                        model = "-mcmodel=kernel";
                                        nolibc = 1;
                                        break;
                                }
                                error(arg);
                                break;
#endif  /* __x86 */
                        case 'M':
                                if (strcmp(arg, "-xM") == 0) {
                                        newae(ctx->i_ae, "-M");
                                        break;
                                }
                                if (strcmp(arg, "-xM1") == 0) {
```

```
                                        newae(ctx->i_ae, "-MM");
                                        break;
                                }
                                error(arg);
                                break;
                        case 'n':
                                if (strcmp(arg, "-xnolib") == 0) {
                                        nolibc = 1;
                                        break;
                                }
                                error(arg);
                                break;
                        case 'O':
                                if (strncmp(arg, "-xO", 3) == 0) {
                                        size_t len = strlen(arg);
                                        char *s;
                                        int c = *(arg + 3);
                                        int level;

                                        if (len != 4 || !isdigit(c))
                                                error(arg);

                                        if ((s = malloc(len)) == NULL)
                                                nomem();

                                        level = atoi(arg + 3);
                                        if (level > 5)
                                                error(arg);
                                        if (level >= 2) {
                                                /*
                                                 * For gcc-3.4.x at -O2 we
                                                 * need to disable optimizations
                                                 * that break ON.
                                                 */
                                                optim_disable(ctx->i_ae, level);
                                                /*
                                                 * limit -xO3 to -O2 as well.
                                                 */
                                                level = 2;
                                        }
                                        (void) snprintf(s, len, "-O%d", level);
                                        newae(ctx->i_ae, s);
                                        free(s);
                                        break;
                                }
                                error(arg);
                                break;
                        case 'p':
                                if (strcmp(arg, "-xpentium") == 0) {
                                        newae(ctx->i_ae, "-march=pentium");
                                        break;
                                }
                                if (strcmp(arg, "-xpg") == 0) {
                                        newae(ctx->i_ae, "-pg");
                                        break;
                                }
                                error(arg);
                                break;
                        case 'r':
                                if (strncmp(arg, "-xregs=", 7) == 0) {
                                        xlate(ctx->i_ae, arg + 7, xregs_tbl);
                                        break;
                                }
                                error(arg);
                                break;
                        case 's':
                                if (strcmp(arg, "-xs") == 0 ||
                                    strcmp(arg, "-xspace") == 0 ||
                                    strcmp(arg, "-xstrconst") == 0)
                                        break;
                                error(arg);
                                break;
                        case ’t’:
                                if (strcmp(arg, "-xtransition") == 0) {
                                        newae(ctx->i_ae, "-Wtransition");
                                        break;
                                }
                                if (strcmp(arg, "-xtrigraphs=yes") == 0) {
                                        newae(ctx->i_ae, "-trigraphs");
                                        break;
                                }
                                if (strcmp(arg, "-xtrigraphs=no") == 0) {
                                        newae(ctx->i_ae, "-notrigraphs");
                                        break;
                                }
                                if (strncmp(arg, "-xtarget=", 9) == 0) {
                                        xlate(ctx->i_ae, arg + 9, xtarget_tbl);
                                        break;
                                }
                                error(arg);
                                break;
                        case ’e’:
                        case ’h’:
                        case ’l’:
                        default:
                                error(arg);
                                break;
                        }
                        break;
                case ’Y’:
                        if (arglen == 1) {
                                if ((arg = *++ctx->i_oldargv) == NULL ||
                                    *arg == ’\0’)
                                        error("-Y");
                                ctx->i_oldargc--;
                                arglen = strlen(arg + 1);
                        } else {
                                arg += 2;
                        }
                        /* Just ignore -YS,... for now */
                        if (strncmp(arg, "S,", 2) == 0)
                                break;
                        if (strncmp(arg, "l,", 2) == 0) {
                                char *s = strdup(arg);
                                s[0] = ’-’;
                                s[1] = ’B’;
                                newae(ctx->i_ae, s);
                                free(s);
                                break;
                        }
                        if (strncmp(arg, "I,", 2) == 0) {
                                char *s = strdup(arg);
                                s[0] = ’-’;
                                s[1] = ’I’;
                                newae(ctx->i_ae, "-nostdinc");
                                newae(ctx->i_ae, s);
                                free(s);
                                break;
                        }
                        error(arg);
                        break;
                case ’Q’:
```

```
                        /*
                         * We could map -Qy into -Wl,-Qy etc.
                         */
                default:
                        error(arg);
                        break;
                }
        }

        if (c_files > 1 && (ctx->i_flags & CW_F_SHADOW) &&
            op != CW_O_PREPROCESS) {
                (void) fprintf(stderr, "%s: error: multiple source files are "
                    "allowed only with -E or -P\n", progname);
                exit(2);
        }

        /*
         * Make sure that we do not have any unintended interactions between
         * the xarch options passed in and the version of the Studio compiler
         * used.
         */
        if ((mflag & (SS11|SS12)) == (SS11|SS12)) {
                (void) fprintf(stderr,
                    "Conflicting \"-xarch=\" flags (both Studio 11 and 12)\n");
                exit(2);
        }

        switch (mflag) {
        case 0:
                /* FALLTHROUGH */
        case M32:
#if defined(__sparc)
                /*
                 * Only -m32 is defined and so put in the missing xarch
                 * translation.
                 */
                newae(ctx->i_ae, "-mcpu=v8");
                newae(ctx->i_ae, "-mno-v8plus");
#endif
                break;
        case M64:
#if defined(__sparc)
                /*
                 * Only -m64 is defined and so put in the missing xarch
                 * translation.
                 */
                newae(ctx->i_ae, "-mcpu=v9");
#endif
                break;
        case SS12:
#if defined(__sparc)
                /* no -m32/-m64 flag used - this is an error for sparc builds */
                (void) fprintf(stderr, "No -m32/-m64 flag defined\n");
                exit(2);
#endif
                break;
        case SS11:
                /* FALLTHROUGH */
        case (SS11|M32):
        case (SS11|M64):
                break;
        case (SS12|M32):
#if defined(__sparc)
                /*
                 * Need to add in further 32 bit options because with SS12
                 * the xarch=sparcvis option can be applied to 32 or 64
                 * bit, and so the translatation table (xtbl) cannot handle
                 * that.
                 */
                newae(ctx->i_ae, "-mv8plus");
#endif
                break;
        case (SS12|M64):
                break;
        default:
                (void) fprintf(stderr,
                    "Incompatible -xarch= and/or -m32/-m64 options used.\n");
                exit(2);
        }
        if (op == CW_O_LINK && (ctx->i_flags & CW_F_SHADOW))
                exit(0);

        if (model && !pic)
                newae(ctx->i_ae, model);
        if (!nolibc)
                newae(ctx->i_ae, "-lc");
        if (!seen_o && (ctx->i_flags & CW_F_SHADOW)) {
                newae(ctx->i_ae, "-o");
                newae(ctx->i_ae, ctx->i_discard);
        }
}
```
______***unchanged_portion_omitted_***

```
**********************************************************
   16093 Wed Oct  1 18:40:29 2014
new/usr/src/uts/intel/Makefile.intel
5196 The cw wrapper restricts gcc to -O2
**********************************************************
# 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) 2012 Nexenta Systems, Inc. All rights reserved.
# Copyright (c) 2013 Andrew Stormont.  All rights reserved.
# Copyright 2014 Gary Mills

#
#       This makefile contains the common definitions for all intel
#       implementation architecture independent modules.
#

#
#       Machine type (implementation architecture):
#
PLATFORM         = i86pc

#
#       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 i86pc
#       because it is the reference port.
#
UNIX_DIR         = $(UTSBASE)/i86pc/unix
GENLIB_DIR       = $(UTSBASE)/intel/genunix
IPDRV_DIR        = $(UTSBASE)/intel/ip
MODSTUBS_DIR     = $(UNIX_DIR)
DSF_DIR          = $(UTSBASE)/$(PLATFORM)/genassym
LINTS_DIR        = $(OBJS_DIR)
LINT_LIB_DIR     = $(UTSBASE)/intel/lint-libs/$(OBJS_DIR)

UNIX_O           = $(UNIX_DIR)/$(OBJS_DIR)/unix.o
GENLIB           = $(GENLIB_DIR)/$(OBJS_DIR)/libgenunix.so
MODSTUBS_O       = $(MODSTUBS_DIR)/$(OBJS_DIR)/modstubs.o
LINT_LIB         = $(UTSBASE)/i86pc/lint-libs/$(OBJS_DIR)/llib-lunix.ln
GEN_LINT_LIB     = $(UTSBASE)/intel/lint-libs/$(OBJS_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 $(UTSBASE)/intel/Makefile.files
include $(UTSBASE)/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

#
#       The following must be defined for all implementations:
#
MODSTUBS                = $(UTSBASE)/intel/ia32/ml/modstubs.s

#
#       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
INLINES_64              = $(UTSBASE)/intel/amd64/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))

COPTFLAG_32             = -_gcc=-fno-toplevel-reorder $(COPTFLAG)
COPTFLAG_64             = -_gcc=-fno-toplevel-reorder $(COPTFLAG64)
COPTFLAG_32             = $(COPTFLAG)
COPTFLAG_64             = $(COPTFLAG64)
COPTIMIZE               = $(COPTFLAG_$(CLASS))

CFLAGS                  = $(CFLAGS_XARCH)
CFLAGS                  += $(COPTIMIZE)
CFLAGS                  += $(INLINES) -D_ASM_INLINES
CFLAGS                  += $(CCMODE)
CFLAGS                  += $(SPACEFLAG)
CFLAGS                  += $(CCUNBOUND)
CFLAGS                  += $(CFLAGS_uts)
CFLAGS                  += -xstrconst

ASFLAGS_XARCH_32        = $(i386_ASFLAGS)
ASFLAGS_XARCH_64        = $(amd64_ASFLAGS)
ASFLAGS_XARCH           = $(ASFLAGS_XARCH_$(CLASS))

ASFLAGS                 += $(ASFLAGS_XARCH)

#
#       Define the base directory for installation.
#
BASE_INS_DIR    = $(ROOT)

#
#       Debugging level
#
#       Special knowledge of which special debugging options affect which
#       file is used to optimize the build if these flags are changed.
#
DEBUG_DEFS_OBJ32        =
DEBUG_DEFS_DBG32        = -DDEBUG
DEBUG_DEFS_OBJ64        =
DEBUG_DEFS_DBG64        = -DDEBUG
DEBUG_DEFS              = $(DEBUG_DEFS_$(BUILD_TYPE))

DEBUG_COND_OBJ32        = $(POUND_SIGN)
DEBUG_COND_DBG32        =
DEBUG_COND_OBJ64        = $(POUND_SIGN)
DEBUG_COND_DBG64        =
IF_DEBUG_OBJ            = $(DEBUG_COND_$(BUILD_TYPE))$(OBJS_DIR)/

$(IF_DEBUG_OBJ)syscall.o        :=      DEBUG_DEFS      += -DSYSCALLTRACE
$(IF_DEBUG_OBJ)clock.o          :=      DEBUG_DEFS      += -DKSLICE=1

#
#       Collect the preprocessor definitions to be associated with *all*
#       files.
#
ALL_DEFS         = $(DEBUG_DEFS) $(OPTION_DEFS)

#
#       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)
#       DRV_KMODS are built both 32-bit and 64-bit
#       DRV_KMODS_32 are built only 32-bit
#       DRV_KMODS_64 are built only 64-bit
#
DRV_KMODS       += aac
DRV_KMODS       += aggr
DRV_KMODS       += ahci
DRV_KMODS       += amd64_gart
DRV_KMODS       += amr
DRV_KMODS       += agpgart
DRV_KMODS       += srn
DRV_KMODS       += agptarget
DRV_KMODS       += arn
DRV_KMODS       += arp
```

```
DRV_KMODS       += asy
DRV_KMODS       += ata
DRV_KMODS       += ath
DRV_KMODS       += atu
DRV_KMODS       += audio
DRV_KMODS       += audio1575
DRV_KMODS       += audio810
DRV_KMODS       += audiocmi
DRV_KMODS       += audiocmihd
DRV_KMODS       += audioemu10k
DRV_KMODS       += audioens
DRV_KMODS       += audiohd
DRV_KMODS       += audioixp
DRV_KMODS       += audiols
DRV_KMODS       += audiop16x
DRV_KMODS       += audiopci
DRV_KMODS       += audiosolo
DRV_KMODS       += audiots
DRV_KMODS       += audiovia823x
DRV_KMODS_32    += audiovia97
DRV_KMODS       += bl
DRV_KMODS       += blkdev
DRV_KMODS       += bge
DRV_KMODS       += bofi
DRV_KMODS       += bpf
DRV_KMODS       += bridge
DRV_KMODS       += bscbus
DRV_KMODS       += bscv
DRV_KMODS       += chxge
DRV_KMODS       += cxgbe
DRV_KMODS       += ntxn
DRV_KMODS       += myri10ge
DRV_KMODS       += clone
DRV_KMODS       += cmdk
DRV_KMODS       += cn
DRV_KMODS       += conskbd
DRV_KMODS       += consms
DRV_KMODS       += cpqary3
DRV_KMODS       += cpuid
DRV_KMODS       += cpunex
DRV_KMODS       += crypto
DRV_KMODS       += cryptoadm
DRV_KMODS       += dca
DRV_KMODS       += devinfo
DRV_KMODS       += dld
DRV_KMODS       += dlpistub
DRV_KMODS_32    += dnet
DRV_KMODS       += dump
DRV_KMODS       += ecpp
DRV_KMODS       += emlxs
DRV_KMODS       += fd
DRV_KMODS       += fdc
DRV_KMODS       += fm
DRV_KMODS       += fssnap
DRV_KMODS       += hxge
DRV_KMODS       += i8042
DRV_KMODS       += i915
DRV_KMODS       += icmp
DRV_KMODS       += icmp6
DRV_KMODS       += intel_nb5000
DRV_KMODS       += intel_nhm
DRV_KMODS       += ip
DRV_KMODS       += ip6
DRV_KMODS       += ipd
DRV_KMODS       += ipf
DRV_KMODS       += ipnet
```

```
DRV_KMODS	+= ippctl
DRV_KMODS	+= ipsecah
DRV_KMODS	+= ipsecesp
DRV_KMODS	+= ipw
DRV_KMODS	+= iwh
DRV_KMODS	+= iwi
DRV_KMODS	+= iwk
DRV_KMODS	+= iwp
DRV_KMODS	+= iwscn
DRV_KMODS	+= kb8042
DRV_KMODS	+= keysock
DRV_KMODS	+= kssl
DRV_KMODS	+= kstat
DRV_KMODS	+= ksyms
DRV_KMODS	+= kmdb
DRV_KMODS	+= llc1
DRV_KMODS	+= lofi
DRV_KMODS	+= log
DRV_KMODS	+= logindmux
DRV_KMODS	+= mega_sas
DRV_KMODS	+= mc-amd
DRV_KMODS	+= mm
DRV_KMODS	+= mouse8042
DRV_KMODS	+= mpt_sas
DRV_KMODS	+= mr_sas
DRV_KMODS	+= mwl
DRV_KMODS	+= nca
DRV_KMODS	+= nsmb
DRV_KMODS	+= nulldriver
DRV_KMODS	+= nv_sata
DRV_KMODS	+= nxge
DRV_KMODS	+= oce
DRV_KMODS	+= openeepr
DRV_KMODS	+= pci_pci
DRV_KMODS	+= pcic
DRV_KMODS	+= pcieb
DRV_KMODS	+= physmem
DRV_KMODS	+= pit_beep
DRV_KMODS	+= pm
DRV_KMODS	+= poll
DRV_KMODS	+= pool
DRV_KMODS	+= power
DRV_KMODS	+= pseudo
DRV_KMODS	+= ptc
DRV_KMODS	+= ptm
DRV_KMODS	+= pts
DRV_KMODS	+= ptsl
DRV_KMODS	+= qlge
DRV_KMODS	+= radeon
DRV_KMODS	+= ral
DRV_KMODS	+= ramdisk
DRV_KMODS	+= random
DRV_KMODS	+= rds
DRV_KMODS	+= rdsv3
DRV_KMODS	+= rpcib
DRV_KMODS	+= rsm
DRV_KMODS	+= rts
DRV_KMODS	+= rtw
DRV_KMODS	+= rum
DRV_KMODS	+= rwd
DRV_KMODS	+= rwn
DRV_KMODS	+= sad
DRV_KMODS	+= sd
DRV_KMODS	+= sdhost
DRV_KMODS	+= sgen
DRV_KMODS	+= si3124
```

```
DRV_KMODS	+= smbios
DRV_KMODS	+= softmac
DRV_KMODS	+= spdsock
DRV_KMODS	+= smbsrv
DRV_KMODS	+= smp
DRV_KMODS	+= sppp
DRV_KMODS	+= sppptun
DRV_KMODS	+= srpt
DRV_KMODS	+= st
DRV_KMODS	+= sy
DRV_KMODS	+= sysevent
DRV_KMODS	+= sysmsg
DRV_KMODS	+= tcp
DRV_KMODS	+= tcp6
DRV_KMODS	+= tl
DRV_KMODS	+= tnf
DRV_KMODS	+= tpm
DRV_KMODS	+= trill
DRV_KMODS	+= udp
DRV_KMODS	+= udp6
DRV_KMODS	+= ucode
DRV_KMODS	+= ural
DRV_KMODS	+= uath
DRV_KMODS	+= urtw
DRV_KMODS	+= vgatext
DRV_KMODS	+= heci
DRV_KMODS	+= vnic
DRV_KMODS	+= vscan
DRV_KMODS	+= wc
DRV_KMODS	+= winlock
DRV_KMODS	+= wpi
DRV_KMODS	+= xge
DRV_KMODS	+= yge
DRV_KMODS	+= zcons
DRV_KMODS	+= zyd
DRV_KMODS	+= simnet
DRV_KMODS	+= stmf
DRV_KMODS	+= stmf_sbd
DRV_KMODS	+= fct
DRV_KMODS	+= fcoe
DRV_KMODS	+= fcoet
DRV_KMODS	+= fcoei
DRV_KMODS	+= qlt
DRV_KMODS	+= iscsit
DRV_KMODS	+= pppt
DRV_KMODS	+= ncall nsctl sdbc nskern sv
DRV_KMODS	+= ii rdc rdcsrv rdcstub
DRV_KMODS	+= iptun

#
# Common code drivers
#

DRV_KMODS	+= afe
DRV_KMODS	+= atge
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

#
# 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

#
#       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):
#
EXEC_KMODS      += elfexec intpexec shbinexec javaexec

#
#       Scheduling Class Modules (/kernel/sched):
#
SCHED_KMODS     += IA RT TS RT_DPTBL TS_DPTBL FSS FX FX_DPTBL SDC

#
#       File System Modules (/kernel/fs):
#
FS_KMODS        += autofs cachefs ctfs dcfs dev devfs fdfs fifofs hsfs lofs
FS_KMODS        += mntfs namefs nfs objfs zfs zut
FS_KMODS        += pcfs procfs sockfs specfs tmpfs udfs ufs sharefs
FS_KMODS        += smbfs

#
#       Streams Modules (/kernel/strmod):
#
STRMOD_KMODS    += bufmod connld dedump ldterm pckt pfmod pipemod
STRMOD_KMODS    += ptem redirmod rpcmod rlmod telmod timod
STRMOD_KMODS    += spppasyn spppcomp
STRMOD_KMODS    += tirdwr ttcompat
STRMOD_KMODS    += usbkbm
STRMOD_KMODS    += usbms
STRMOD_KMODS    += usbwcm
STRMOD_KMODS    += usb_ah
STRMOD_KMODS    += drcompat
STRMOD_KMODS    += cryptmod
STRMOD_KMODS    += vuid2ps2
STRMOD_KMODS    += vuid3ps2
STRMOD_KMODS    += vuidm3p
STRMOD_KMODS    += vuidm4p
STRMOD_KMODS    += vuidm5p

#
#       'System' Modules (/kernel/sys):
#
SYS_KMODS       += c2audit
SYS_KMODS       += doorfs
SYS_KMODS       += exacctsys
SYS_KMODS       += inst_sync
SYS_KMODS       += kaio
SYS_KMODS       += msgsys
SYS_KMODS       += pipe
SYS_KMODS       += portfs
SYS_KMODS       += pset
SYS_KMODS       += semsys
SYS_KMODS       += shmsys
SYS_KMODS       += sysacct
SYS_KMODS       += acctctl

#
#       'Misc' Modules (/kernel/misc)
#       MISC_KMODS are built both 32-bit and 64-bit
#       MISC_KMODS_32 are built only 32-bit
#       MISC_KMODS_64 are built only 64-bit
#
MISC_KMODS      += ac97
MISC_KMODS      += acpica
MISC_KMODS      += agpmaster
MISC_KMODS      += bignum
MISC_KMODS      += bootdev
MISC_KMODS      += busra
MISC_KMODS      += cmlb
MISC_KMODS      += consconfig
MISC_KMODS      += ctf
MISC_KMODS      += dadk
MISC_KMODS      += dcopy
MISC_KMODS      += dls
MISC_KMODS      += drm
MISC_KMODS      += fssnap_if
MISC_KMODS      += gda
MISC_KMODS      += gld
MISC_KMODS      += hidparser
MISC_KMODS      += hook
MISC_KMODS      += hpcsvc
MISC_KMODS      += ibcm
MISC_KMODS      += ibdm
```

```
MISC_KMODS      += ibdma
MISC_KMODS      += ibmf
MISC_KMODS      += ibtl
MISC_KMODS      += idm
MISC_KMODS      += idmap
MISC_KMODS      += iommulib
MISC_KMODS      += ipc
MISC_KMODS      += kbtrans
MISC_KMODS      += kcf
MISC_KMODS      += kgssapi
MISC_KMODS      += kmech_dummy
MISC_KMODS      += kmech_krb5
MISC_KMODS      += ksocket
MISC_KMODS      += mac
MISC_KMODS      += mii
MISC_KMODS      += mwlfw
MISC_KMODS      += net80211
MISC_KMODS      += nfs_dlboot
MISC_KMODS      += nfssrv
MISC_KMODS      += neti
MISC_KMODS      += pci_autoconfig
MISC_KMODS      += pcicfg
MISC_KMODS      += pcihp
MISC_KMODS      += pcmcia
MISC_KMODS      += rpcsec
MISC_KMODS      += rpcsec_gss
MISC_KMODS      += rsmops
MISC_KMODS      += sata
MISC_KMODS      += scsi
MISC_KMODS      += sda
MISC_KMODS      += sol_ofs
MISC_KMODS      += spuni
MISC_KMODS      += strategy
MISC_KMODS      += strplumb
MISC_KMODS      += tem
MISC_KMODS      += tlimod
MISC_KMODS      += usba usba10 usbs49_fw
MISC_KMODS      += scsi_vhci_f_sym_hds
MISC_KMODS      += scsi_vhci_f_sym
MISC_KMODS      += scsi_vhci_f_tpgs
MISC_KMODS      += scsi_vhci_f_asym_sun
MISC_KMODS      += scsi_vhci_f_tape
MISC_KMODS      += scsi_vhci_f_tpgs_tape
MISC_KMODS      += fctl
MISC_KMODS      += emlxs_fw
MISC_KMODS      += qlc_fw_2200
MISC_KMODS      += qlc_fw_2300
MISC_KMODS      += qlc_fw_2400
MISC_KMODS      += qlc_fw_2500
MISC_KMODS      += qlc_fw_6322
MISC_KMODS      += qlc_fw_8100
MISC_KMODS      += hwa1480_fw
MISC_KMODS      += uathfw
MISC_KMODS      += uwba

MISC_KMODS      += klmmod klmops

#
#       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

#
#	Modules eXcluded from the product:
#

#
#	'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

#
# Ensure that the variable member of the cpu_t (cpu_m) is defined
# for the lint builds so as not to cause lint errors during the
```

```
# global cross check.
#
LINTFLAGS	+= -D_MACHDEP -I$(UTSBASE)/i86pc
```