```
**********************************************************
   35460 Mon Jun 25 17:12:19 2012
new/usr/src/Makefile.master
include build stamp in unix
**********************************************************
```

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

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

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

#
# RELEASE_BUILD should be cleared for final release builds.
# NOT_RELEASE_BUILD is exactly what the name implies.
#
# INTERNAL_RELEASE_BUILD is a subset of RELEASE_BUILD. It mostly controls
# identification strings. Enabling RELEASE_BUILD automatically enables
# INTERNAL_RELEASE_BUILD.
#
# EXPORT_RELEASE_BUILD controls whether binaries are built in a form that
# can be released for export under a binary license.  It is orthogonal to
# the other *RELEASE_BUILD settings.  ("#" means do an export release
# build, "" means do a normal build.)
#
# CLOSED_BUILD controls whether we try to build files under
# usr/closed.  ("" means to build closed code, "#" means don't try to
# build it.)  Skipping the closed code implies doing an export release
# build.
#
# STRIP_COMMENTS toggles comment section striping. Generally the same setting
# as INTERNAL_RELEASE_BUILD.
#
# __GNUC toggles the building of ON components using gcc and related tools.
# Normally set to '#', set it to '' to do gcc build.
#
# The declaration POUND_SIGN is always '#'. This is needed to get around the
# make feature that '#' is always a comment delimiter, even when escaped or
# quoted. We use this macro expansion method to get POUND_SIGN rather than
# always breaking out a shell because the general case can cause a noticable
# slowdown in build times when so many Makefiles include Makefile.master.
#
```



```
# While the majority of users are expected to override the setting below
# with an env file (via nightly or bldenv), if you aren't building that way
# (ie, you're using "ws" or some other bootstrapping method) then you need
# this definition in order to avoid the subshell invocation mentioned above.
#

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

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

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

EXPORT_RELEASE_BUILD=                   $(POUND_SIGN)
$(CLOSED_BUILD)EXPORT_RELEASE_BUILD=

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

STRIP_COMMENTS= $(INTERNAL_RELEASE_BUILD)

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


# The variables below control the compilers used during the build.
# There are a number of permutations.
#
# __GNUC and __SUNC control (and indicate) the primary compiler.  Whichever
# one is not POUND_SIGN is the primary, with the other as the shadow.  They
# may also be used to control entirely compiler-specific Makefile assignments.
# __SUNC and Sun Studio are the default.
#
# __GNUC64 indicates that the 64bit build should use the GNU C compiler.
# There is no Sun C analogue.
#
# The following version-specific options are operative regardless of which
# compiler is primary, and control the versions of the given compilers to be
# used.  They also allow compiler-version specific Makefile fragments.
#
# __SSNEXT when set to the empty string enables options specific to the 'next'
# version of the Sun Studio compiler.
#
# __GNUC3 when the empty string uses and refers to GCC 3.x, it is the default.
# __GNUC4 when the empty string uses and refers to GCC 4.x.

__GNUC=                 $(POUND_SIGN)
$(__GNUC)__SUNC=        $(POUND_SIGN)
__GNUC64=               $(__GNUC)
```

```
__SSNEXT=               $(POUND_SIGN)

__GNUC3=
__GNUC4=                $(POUND_SIGN)
$(__GNUC4)__GNUC3=      $(POUND_SIGN)

# 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)

$(__GNUC3)GCC_ROOT=     $(SFW_ROOT)
$(__GNUC4)GCC_ROOT=     /opt/gcc/4.4.4
GCCLIBDIR=      $(GCC_ROOT)/lib
GCCLIBDIR64=    $(GCC_ROOT)/lib/$(MACH64)

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
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
BUILDSTAMP=     $(ONBLD_TOOLS)/bin/buildstamp
FIND=           /usr/bin/find
PERL=           /usr/bin/perl
PYTHON_24=      /usr/bin/python2.4
PYTHON_26=      /usr/bin/python2.6
PYTHON=         $(PYTHON_24)
SORT=           /usr/bin/sort
TOUCH=          /usr/bin/touch
WC=             /usr/bin/wc
XARGS=          /usr/bin/xargs
ELFEDIT=        /usr/bin/elfedit
ELFSIGN=        /usr/bin/elfsign
DTRACE=         /usr/sbin/dtrace -xnolibs
UNIQ=           /usr/bin/uniq
TAR=            /usr/bin/tar

FILEMODE=       644
DIRMODE=        755

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

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

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

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

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

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

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

#
# IMPORTANT:: If you change any of INS.file, INS.dir, INS.rename,
```

```
260 # INS.link or INS.symlink here, then you must also change the
261 # corresponding override definitions in $CLOSED/Makefile.tonic.
262 # If you do not do this, then the closedbins build for the OpenSolaris
263 # community will break. PS, the gatekeepers will be upset too.
264 INS.file=       $(RM) $@; $(INS) -s -m $(FILEMODE) -f $(@D) $<
265 INS.dir=        $(INS) -s -d -m $(DIRMODE) $@
266 # installs and renames at once
267 #
268 INS.rename=     $(INS.file); $(MV) $(@D)/$(<F) $@

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

275 #
276 # Python bakes the mtime of the .py file into the compiled .pyc and
277 # rebuilds if the baked-in mtime != the mtime of the source file
278 # (rather than only if it's less than), thus when installing python
279 # files we must make certain to not adjust the mtime of the source
280 # (.py) file.
281 #
282 INS.pyfile=     $(INS.file); $(TOUCH) -r $< $@

284 # MACH must be set in the shell environment per uname -p on the build host
285 # More specific architecture variables should be set in lower makefiles.
286 #
287 # MACH64 is derived from MACH, and BUILD64 is set to '#' for
288 # architectures on which we do not build 64-bit versions.
289 # (There are no such architectures at the moment.)
290 #
291 # Set BUILD64=# in the environment to disable 64-bit amd64
292 # builds on i386 machines.

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

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

300 sparc_BUILD64=
301 i386_BUILD64=
302 BUILD64=        $($(MACH)_BUILD64)

304 #
305 # C compiler mode. Future compilers may change the default on us,
306 # so force extended ANSI mode globally. Lower level makefiles can
307 # override this by setting CCMODE.
308 #
309 CCMODE=                 -Xa
310 CCMODE64=               -Xa

312 #
313 # C compiler verbose mode. This is so we can enable it globally,
314 # but turn it off in the lower level makefiles of things we cannot
315 # (or aren't going to) fix.
316 #
317 CCVERBOSE=              -v

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

323 # set this to the secret flag "-Wc,-Qiselect-regsym=0" to disable register
324 # symbols (used to detect conflicts between objects that use global registers)
325 # we disable this now for safety, and because genunix doesn't link with
```



```
326 # this feature (the v9 default) enabled.
327 #
328 # REGSYM is separate since the C++ driver syntax is different.
329 CCREGSYM=               -Wc,-Qiselect-regsym=0
330 CCCREGSYM=              -Qoption cg -Qiselect-regsym=0

332 # Prevent the removal of static symbols by the SPARC code generator (cg).
333 # The x86 code generator (ube) does not remove such symbols and as such
334 # using this workaround is not applicable for x86.
335 #
336 CCSTATICSYM=            -Wc,-Qassembler-ounrefsym=0
337 #
338 # generate 32-bit addresses in the v9 kernel. Saves memory.
339 CCABS32=                -Wc,-xcode=abs32
340 #
341 # generate v9 code which tolerates callers using the v7 ABI, for the sake of
342 # system calls.
343 $(__GNUC4)CC32BITCALLERS=               -_gcc=-massume-32bit-callers

345 # GCC, especially, is increasingly beginning to auto-inline functions and
346 # sadly does so separately not under the general -fno-inline-functions
347 $(__GNUC4)CCNOAUTOINLINE= -_gcc=-fno-inline-small-functions \
348         -_gcc=-fno-inline-functions-called-once

350 # One optimization the compiler might perform is to turn this:
351 #       #pragma weak foo
352 #       extern int foo;
353 #       if (&foo)
354 #               foo = 5;
355 # into
356 #       foo = 5;
357 # Since we do some of this (foo might be referenced in common kernel code
358 # but provided only for some cpu modules or platforms), we disable this
359 # optimization.
360 #
361 sparc_CCUNBOUND = -Wd,-xsafe=unboundsym
362 i386_CCUNBOUND  =
363 CCUNBOUND       = $($(MACH)_CCUNBOUND)

365 #
366 # compiler '-xarch' flag. This is here to centralize it and make it
367 # overridable for testing.
368 sparc_XARCH=    -m32
369 sparcv9_XARCH=  -m64
370 i386_XARCH=
371 amd64_XARCH=    -m64 -Ui386 -U__i386

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

379 #
380 # These flags define what we need to be 'standalone' i.e. -not- part
381 # of the rather more cosy userland environment.  This basically means
382 # the kernel.
383 #
384 # XX64  future versions of gcc will make -mcmodel=kernel imply -mno-red-zone
385 #
386 sparc_STAND_FLAGS=      -_gcc=-ffreestanding
387 sparcv9_STAND_FLAGS=    -_gcc=-ffreestanding
388 i386_STAND_FLAGS=       -_gcc=-ffreestanding
389 amd64_STAND_FLAGS=      -xmodel=kernel

391 SAVEARGS=               -Wu,-save_args
```

```
392 amd64_STAND_FLAGS       += $(SAVEARGS)

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

397 #
398 # disable the incremental linker
399 ILDOFF=                 -xildoff
400 #
401 XDEPEND=                -xdepend
402 XFFLAG=                 -xF=%all
403 XESS=                   -xs
404 XSTRCONST=              -xstrconst

406 #
407 # turn warnings into errors (C)
408 CERRWARN = -errtags=yes -errwarn=%all
409 CERRWARN += -erroff=E_EMPTY_TRANSLATION_UNIT
410 CERRWARN += -erroff=E_STATEMENT_NOT_REACHED

412 # Normally cw(1) would translate -v into a set of options including these
413 # but as they're GCC 4.x specific, we can't do that
414 $(__GNUC4)CERRWARN +=   -_gcc=-Wno-address -_gcc=-Wno-array-bounds

416 #
417 # turn warnings into errors (C++)
418 CCERRWARN=              -xwe

420 # C99 mode
421 C99_ENABLE=     -xc99=%all
422 C99_DISABLE=    -xc99=%none
423 C99MODE=        $(C99_DISABLE)
424 C99LMODE=       $(C99MODE:-xc99%=-Xc99%)

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

434 sparc_ASFLAGS=  $(sparc_AS_XARCH)
435 sparcv9_ASFLAGS=$(sparcv9_AS_XARCH)
436 i386_ASFLAGS=   $(i386_AS_XARCH)
437 amd64_ASFLAGS=  $(amd64_AS_XARCH)

439 #
440 sparc_COPTFLAG=         -xO3
441 sparcv9_COPTFLAG=       -xO3
442 i386_COPTFLAG=          -O
443 amd64_COPTFLAG=         -xO3

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

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

452 # Direct the Sun Studio compiler to use a static globalization prefix based on t
453 # name of the module rather than something unique. Otherwise, objects
454 # will not build deterministically, as subsequent compilations of identical
455 # source will yeild objects that always look different.
456 #
457 # In the same spirit, this will also remove the date from the N_OPT stab.
```



```
458 CGLOBALSTATIC= -W0,-xglobalstatic

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

464 #
465 # Default debug format for Sun Studio 11 is dwarf, so force it to
466 # generate stabs.
467 #
468 DEBUGFORMAT=    -xdebugformat=stabs

470 #
471 # Flags used to build in debug mode for ctf generation.  Bugs in the Devpro
472 # compilers currently prevent us from building with cc-emitted DWARF.
473 #
474 CTF_FLAGS_sparc = -g -Wc,-Qiselect-T1 $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)
475 CTF_FLAGS_i386  = -g $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)
476 CTF_FLAGS       = $(CTF_FLAGS_$(MACH)) $(DEBUGFORMAT)

478 #
479 # Flags used with genoffsets
480 #
481 GOFLAGS = -_noecho \
482         $(CALLSYMS) \
483         $(CDWARFSTR)

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

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

491 #
492 # tradeoff time for space (smaller is better)
493 #
494 sparc_SPACEFLAG         = -xspace -W0,-Lt
495 sparcv9_SPACEFLAG       = -xspace -W0,-Lt
496 i386_SPACEFLAG          = -xspace
497 amd64_SPACEFLAG         =

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

502 #
503 # The Sun Studio 11 compiler has changed the behaviour of integer
504 # wrap arounds and so a flag is needed to use the legacy behaviour
505 # (without this flag panics/hangs could be exposed within the source).
506 #
507 sparc_IROPTFLAG         = -W2,-xwrap_int
508 sparcv9_IROPTFLAG       = -W2,-xwrap_int
509 i386_IROPTFLAG          =
510 amd64_IROPTFLAG         =

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

515 sparc_XREGSFLAG         = -xregs=no%appl
516 sparcv9_XREGSFLAG       = -xregs=no%appl
517 i386_XREGSFLAG          =
518 amd64_XREGSFLAG         =

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

523 CFLAGS=         $(COPTFLAG) $($(MACH)_CFLAGS) $(SPACEFLAG) $(CCMODE) \
```

```
524                 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG) \
525                 $(CGLOBALSTATIC) $(CCNOAUTOINLINE)
526 CFLAGS64=       $(COPTFLAG64) $($(MACH64)_CFLAGS) $(SPACEFLAG64) $(CCMODE64) \
527                 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG64) \
528                 $(CGLOBALSTATIC) $(CCNOAUTOINLINE)
529 #
530 # Flags that are used to build parts of the code that are subsequently
531 # run on the build machine (also known as the NATIVE_BUILD).
532 #
533 NATIVE_CFLAGS=  $(COPTFLAG) $($(NATIVE_MACH)_CFLAGS) $(CCMODE) \
534                 $(ILDOFF) $(CERRWARN) $(C99MODE) $($(NATIVE_MACH)_CCUNBOUND) \
535                 $(IROPTFLAG) $(CGLOBALSTATIC) $(CCNOAUTOINLINE)

537 DTEXTDOM=-DTEXT_DOMAIN=\"$(TEXT_DOMAIN)\"       # For messaging.
538 DTS_ERRNO=-D_TS_ERRNO
539 CPPFLAGS.master=$(DTEXTDOM) $(DTS_ERRNO) \
540         $(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) $(ENVCPPFLAGS4)
541 CPPFLAGS.native=$(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) $(ENVCPPFLAGS4)
542 CPPFLAGS=       $(CPPFLAGS.master)
543 AS_CPPFLAGS=    $(CPPFLAGS.master)
544 JAVAFLAGS=      -deprecation

546 #
547 # For source message catalogue
548 #
549 .SUFFIXES: $(SUFFIXES) .i .po
550 MSGROOT= $(ROOT)/catalog
551 MSGDOMAIN= $(MSGROOT)/$(TEXT_DOMAIN)
552 MSGDOMAINPOFILE = $(MSGDOMAIN)/$(POFILE)
553 DCMSGDOMAIN= $(MSGROOT)/LC_TIME/$(TEXT_DOMAIN)
554 DCMSGDOMAINPOFILE = $(DCMSGDOMAIN)/$(DCFILE:.dc=.po)

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

568 #
569 # This is overwritten by local Makefile when PROG is a list.
570 #
571 POFILE= $(PROG).po

573 sparc_CCFLAGS=          -cg92 -compat=4 \
574                         -Qoption ccfe -messages=no%anachronism \
575                         $(CCERRWARN)
576 sparcv9_CCFLAGS=        $(sparcv9_XARCH) -dalign -compat=5 \
577                         -Qoption ccfe -messages=no%anachronism \
578                         -Qoption ccfe -features=no%conststrings \
579                         $(CCCREGSYM) \
580                         $(CCERRWARN)
581 i386_CCFLAGS=           -compat=4 \
582                         -Qoption ccfe -messages=no%anachronism \
583                         -Qoption ccfe -features=no%conststrings \
584                         $(CCERRWARN)
585 amd64_CCFLAGS=          $(amd64_XARCH) -compat=5 \
586                         -Qoption ccfe -messages=no%anachronism \
587                         -Qoption ccfe -features=no%conststrings \
588                         $(CCERRWARN)
```



```
590 sparc_CCOPTFLAG=        -O
591 sparcv9_CCOPTFLAG=      -O
592 i386_CCOPTFLAG=         -O
593 amd64_CCOPTFLAG=        -O

595 CCOPTFLAG=      $($(MACH)_CCOPTFLAG)
596 CCOPTFLAG64=    $($(MACH64)_CCOPTFLAG)
597 CCFLAGS=        $(CCOPTFLAG) $($(MACH)_CCFLAGS)
598 CCFLAGS64=      $(CCOPTFLAG64) $($(MACH64)_CCFLAGS)

600 #
601 #
602 #
603 ELFWRAP_FLAGS   =
604 ELFWRAP_FLAGS64 =       -64

606 #
607 # Various mapfiles that are used throughout the build, and delivered to
608 # /usr/lib/ld.
609 #
610 MAPFILE.NED_i386 =      $(SRC)/common/mapfiles/common/map.noexdata
611 MAPFILE.NED_sparc =
612 MAPFILE.NED =           $(MAPFILE.NED_$(MACH))
613 MAPFILE.PGA =           $(SRC)/common/mapfiles/common/map.pagealign
614 MAPFILE.NES =           $(SRC)/common/mapfiles/common/map.noexstk
615 MAPFILE.FLT =           $(SRC)/common/mapfiles/common/map.filter
616 MAPFILE.LEX =           $(SRC)/common/mapfiles/common/map.lex.yy

618 #
619 # Generated mapfiles that are compiler specific, and used throughout the
620 # build.  These mapfiles are not delivered in /usr/lib/ld.
621 #
622 MAPFILE.NGB_sparc=      $(SRC)/common/mapfiles/gen/sparc_cc_map.noexeglobs
623 $(__GNUC64)MAPFILE.NGB_sparc= \
624                         $(SRC)/common/mapfiles/gen/sparc_gcc_map.noexeglobs
625 MAPFILE.NGB_sparcv9=    $(SRC)/common/mapfiles/gen/sparcv9_cc_map.noexeglobs
626 $(__GNUC64)MAPFILE.NGB_sparcv9= \
627                         $(SRC)/common/mapfiles/gen/sparcv9_gcc_map.noexeglobs
628 MAPFILE.NGB_i386=       $(SRC)/common/mapfiles/gen/i386_cc_map.noexeglobs
629 $(__GNUC64)MAPFILE.NGB_i386= \
630                         $(SRC)/common/mapfiles/gen/i386_gcc_map.noexeglobs
631 MAPFILE.NGB_amd64=      $(SRC)/common/mapfiles/gen/amd64_cc_map.noexeglobs
632 $(__GNUC64)MAPFILE.NGB_amd64= \
633                         $(SRC)/common/mapfiles/gen/amd64_gcc_map.noexeglobs
634 MAPFILE.NGB =           $(MAPFILE.NGB_$(MACH))

636 #
637 # A generic interface mapfile name, used by various dynamic objects to define
638 # the interfaces and interposers the object must export.
639 #
640 MAPFILE.INT =           mapfile-intf

642 #
643 # LDLIBS32 can be set in the environment to override the following assignment.
644 # LDLIBS64 can be set to override the assignment made in Makefile.master.64.
645 # These environment settings make sure that no libraries are searched outside
646 # of the local workspace proto area:
647 #       LDLIBS32=-YP,$ROOT/lib:$ROOT/usr/lib
648 #       LDLIBS64=-YP,$ROOT/lib/$MACH64:$ROOT/usr/lib/$MACH64
649 #
650 LDLIBS32 =      $(ENVLDLIBS1) $(ENVLDLIBS2) $(ENVLDLIBS3)
651 LDLIBS.cmd =    $(LDLIBS32)
652 LDLIBS.lib =    $(LDLIBS32)
653 #
654 # Define compilation macros.
655 #
```

```
656 COMPILE.c=      $(CC) $(CFLAGS) $(CPPFLAGS) -c
657 COMPILE64.c=    $(CC) $(CFLAGS64) $(CPPFLAGS) -c
658 COMPILE.cc=     $(CCC) $(CCFLAGS) $(CPPFLAGS) -c
659 COMPILE64.cc=   $(CCC) $(CCFLAGS64) $(CPPFLAGS) -c
660 COMPILE.s=      $(AS) $(ASFLAGS) $(AS_CPPFLAGS)
661 COMPILE64.s=    $(AS) $(ASFLAGS) $($(MACH64)_AS_XARCH) $(AS_CPPFLAGS)
662 COMPILE.d=      $(DTRACE) -G -32
663 COMPILE64.d=    $(DTRACE) -G -64
664 COMPILE.b=      $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))
665 COMPILE64.b=    $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))

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

670 #
671 # Link time macros
672 #
673 CCNEEDED                = -lC
674 CCEXTNEEDED             = -lCrun -lCstd
675 $(__GNUC)CCNEEDED       = -L$(GCCLIBDIR) -R$(GCCLIBDIR) -lstdc++ -lgcc_s
676 $(__GNUC)CCEXTNEEDED    = $(CCNEEDED)

678 LINK.c=         $(CC) $(CFLAGS) $(CPPFLAGS) $(LDFLAGS)
679 LINK64.c=       $(CC) $(CFLAGS64) $(CPPFLAGS) $(LDFLAGS)
680 NORUNPATH=      -norunpath -nolib
681 LINK.cc=        $(CCC) $(CCFLAGS) $(CPPFLAGS) $(NORUNPATH) \
682                 $(LDFLAGS) $(CCNEEDED)
683 LINK64.cc=      $(CCC) $(CCFLAGS64) $(CPPFLAGS) $(NORUNPATH) \
684                 $(LDFLAGS) $(CCNEEDED)

686 #
687 # lint macros
688 #
689 # Note that the undefine of __PRAGMA_REDEFINE_EXTNAME can be removed once
690 # ON is built with a version of lint that has the fix for 4484186.
691 #
692 ALWAYS_LINT_DEFS =      -errtags=yes -s
693 ALWAYS_LINT_DEFS +=     -erroff=E_PTRDIFF_OVERFLOW
694 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_NARROW_CONV
695 ALWAYS_LINT_DEFS +=     -U__PRAGMA_REDEFINE_EXTNAME
696 ALWAYS_LINT_DEFS +=     $(C99LMODE)
697 ALWAYS_LINT_DEFS +=     -errsecurity=$(SECLEVEL)
698 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_CREAT_WITHOUT_EXCL
699 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_FORBIDDEN_WARN_CREAT
700 # XX64 -- really only needed for amd64 lint
701 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_INT_TO_SMALL_INT
702 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_CONST_TO_SMALL_INT
703 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_TO_SMALL_INT
704 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_TO_PTR_FROM_INT
705 ALWAYS_LINT_DEFS +=     -erroff=E_COMP_INT_WITH_LARGE_INT
706 ALWAYS_LINT_DEFS +=     -erroff=E_INTEGRAL_CONST_EXP_EXPECTED
707 ALWAYS_LINT_DEFS +=     -erroff=E_PASS_INT_TO_SMALL_INT
708 ALWAYS_LINT_DEFS +=     -erroff=E_PTR_CONV_LOSES_BITS

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

714 SECLEVEL=       core
715 LINT.c=         $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS) $(CPPFLAGS) \
716                 $(ALWAYS_LINT_DEFS)
717 LINT64.c=       $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS64) $(CPPFLAGS) \
718                 $(ALWAYS_LINT_DEFS)
719 LINT.s=         $(LINT.c)

721 # For some future builds, NATIVE_MACH and MACH might be different.
```



```
722 # Therefore, NATIVE_MACH needs to be redefined in the
723 # environment as 'uname -p' to override this macro.
724 #
725 # For now at least, we cross-compile amd64 on i386 machines.
726 NATIVE_MACH=    $(MACH:amd64=i386)

728 # Define native compilation macros
729 #

731 # Base directory where compilers are loaded.
732 # Defined here so it can be overridden by developer.
733 #
734 SPRO_ROOT=              $(BUILD_TOOLS)/SUNWspro
735 SPRO_VROOT=             $(SPRO_ROOT)/SS12
736 GNU_ROOT=               $(SFW_ROOT)

738 # Till SS12u1 formally becomes the NV CBE, LINT is hard
739 # coded to be picked up from the $SPRO_ROOT/sunstudio12.1/
740 # location. Impacted variables are sparc_LINT, sparcv9_LINT,
741 # i386_LINT, amd64_LINT.
742 # Reset them when SS12u1 is rolled out.
743 #

745 # Specify platform compiler versions for languages
746 # that we use (currently only c and c++).
747 #
748 sparc_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
749 $(__GNUC)sparc_CC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
750 sparc_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
751 $(__GNUC)sparc_CCC=     $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
752 sparc_CPP=              /usr/ccs/lib/cpp
753 sparc_AS=               /usr/ccs/bin/as -xregsym=no
754 sparc_LD=               /usr/ccs/bin/ld
755 sparc_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

757 sparcv9_CC=            $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
758 $(__GNUC64)sparcv9_CC=  $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
759 sparcv9_CCC=           $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
760 $(__GNUC64)sparcv9_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
761 sparcv9_CPP=           /usr/ccs/lib/cpp
762 sparcv9_AS=            /usr/ccs/bin/as -xregsym=no
763 sparcv9_LD=            /usr/ccs/bin/ld
764 sparcv9_LINT=          $(SPRO_ROOT)/sunstudio12.1/bin/lint

766 i386_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
767 $(__GNUC)i386_CC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
768 i386_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
769 $(__GNUC)i386_CCC=     $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
770 i386_CPP=              /usr/ccs/lib/cpp
771 i386_AS=               /usr/ccs/bin/as
772 $(__GNUC)i386_AS=      $(ONBLD_TOOLS)/bin/$(MACH)/aw
773 i386_LD=               /usr/ccs/bin/ld
774 i386_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

776 amd64_CC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
777 $(__GNUC64)amd64_CC=   $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
778 amd64_CCC=             $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
779 $(__GNUC64)amd64_CCC=  $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
780 amd64_CPP=             /usr/ccs/lib/cpp
781 amd64_AS=              $(ONBLD_TOOLS)/bin/$(MACH)/aw
782 amd64_LD=              /usr/ccs/bin/ld
783 amd64_LINT=            $(SPRO_ROOT)/sunstudio12.1/bin/lint

785 NATIVECC=             $($(NATIVE_MACH)_CC)
786 NATIVECCC=            $($(NATIVE_MACH)_CCC)
787 NATIVECPP=            $($(NATIVE_MACH)_CPP)
```

```
788 NATIVEAS=               $($(NATIVE_MACH)_AS)
789 NATIVELD=               $($(NATIVE_MACH)_LD)
790 NATIVELINT=            $($(NATIVE_MACH)_LINT)

792 #
793 # Makefile.master.64 overrides these settings
794 #
795 CC=                     $(NATIVECC)
796 CCC=                    $(NATIVECCC)
797 CPP=                    $(NATIVECPP)
798 AS=                     $(NATIVEAS)
799 LD=                     $(NATIVELD)
800 LINT=                   $(NATIVELINT)

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

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

811 BDIRECT=        -Bdirect
812 BDYNAMIC=       -Bdynamic
813 BLOCAL=         -Blocal
814 BNODIRECT=      -Bnodirect
815 BREDUCE=        -Breduce
816 BSTATIC=        -Bstatic

818 ZDEFS=          -zdefs
819 ZDIRECT=        -zdirect
820 ZIGNORE=        -zignore
821 ZINITFIRST=     -zinitfirst
822 ZINTERPOSE=     -zinterpose
823 ZLAZYLOAD=      -zlazyload
824 ZLOADFLTR=      -zloadfltr
825 ZMULDEFS=       -zmuldefs
826 ZNODEFAULTLIB=  -znodefaultlib
827 ZNODEFS=        -znodefs
828 ZNODELETE=      -znodelete
829 ZNODLOPEN=      -znodlopen
830 ZNODUMP=        -znodump
831 ZNOLAZYLOAD=    -znolazyload
832 ZNOLDYNSYM=     -znoldynsym
833 ZNORELOC=       -znoreloc
834 ZNOVERSION=     -znoversion
835 ZRECORD=        -zrecord
836 ZREDLOCSYM=     -zredlocsym
837 ZTEXT=          -ztext
838 ZVERBOSE=       -zverbose

840 GSHARED=        -G
841 CCMT=           -mt

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

846 sparc_C_PICFLAGS =      -K pic
847 sparcv9_C_PICFLAGS =    -K pic
848 i386_C_PICFLAGS =       -K pic
849 amd64_C_PICFLAGS =      -K pic
850 C_PICFLAGS =            $($(MACH)_C_PICFLAGS)
851 C_PICFLAGS64 =          $($(MACH64)_C_PICFLAGS)

853 sparc_C_BIGPICFLAGS =   -K PIC
```



```
854 sparcv9_C_BIGPICFLAGS = -K PIC
855 i386_C_BIGPICFLAGS =    -K PIC
856 amd64_C_BIGPICFLAGS =   -K PIC
857 C_BIGPICFLAGS =         $($(MACH)_C_BIGPICFLAGS)
858 C_BIGPICFLAGS64 =       $($(MACH64)_C_BIGPICFLAGS)

860 # CC requires there to be no space between '-K' and 'pic' or 'PIC'.
861 sparc_CC_PICFLAGS =     -Kpic
862 sparcv9_CC_PICFLAGS =   -KPIC
863 i386_CC_PICFLAGS =      -Kpic
864 amd64_CC_PICFLAGS =     -Kpic
865 CC_PICFLAGS =           $($(MACH)_CC_PICFLAGS)
866 CC_PICFLAGS64 =         $($(MACH64)_CC_PICFLAGS)

868 AS_PICFLAGS=            $(C_PICFLAGS)
869 AS_BIGPICFLAGS=         $(C_BIGPICFLAGS)

871 #
872 # Default label for CTF sections
873 #
874 CTFCVTFLAGS=            -i -L VERSION

876 #
877 # Override to pass module-specific flags to ctfmerge.  Currently used
878 # only by krtld to turn on fuzzy matching.
879 #
880 CTFMRGFLAGS=

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

884 ELFSIGN_O=      $(TRUE)
885 ELFSIGN_CRYPTO= $(ELFSIGN_O)
886 ELFSIGN_OBJECT= $(ELFSIGN_O)
887 $(EXPORT_RELEASE_BUILD)ELFSIGN_O =      $(ELFSIGN)
888 $(EXPORT_RELEASE_BUILD)ELFSIGN_CFNAME = SUNWosnetCF
889 $(EXPORT_RELEASE_BUILD)ELFSIGN_KEY =    \
890                         $(CLOSED)/cmd/cmd-crypto/etc/keys/$(ELFSIGN_CFNAME)
891 $(EXPORT_RELEASE_BUILD)ELFSIGN_CERT=    \
892                         $(CLOSED)/cmd/cmd-crypto/etc/certs/$(ELFSIGN_CFNAME)
893 $(EXPORT_RELEASE_BUILD)ELFSIGN_SENAME = SUNWosnetSE
894 $(EXPORT_RELEASE_BUILD)ELFSIGN_SEKEY =  \
895                         $(CLOSED)/cmd/cmd-crypto/etc/keys/$(ELFSIGN_SENAME)
896 $(EXPORT_RELEASE_BUILD)ELFSIGN_SECERT=  \
897                         $(CLOSED)/cmd/cmd-crypto/etc/certs/$(ELFSIGN_SENAME)
898 $(EXPORT_RELEASE_BUILD)ELFSIGN_CRYPTO=  $(ELFSIGN_O) sign \
899                         $(ELFSIGN_FORMAT_OPTION) \
900                         -k $(ELFSIGN_KEY) -c $(ELFSIGN_CERT) -e $@
901 $(EXPORT_RELEASE_BUILD)ELFSIGN_OBJECT=  $(ELFSIGN_O) sign \
902                         $(ELFSIGN_FORMAT_OPTION) \
903                         -k $(ELFSIGN_SEKEY) -c $(ELFSIGN_SECERT) -e $@

905 # Rules (normally from make.rules) and macros which are used for post
906 # processing files. Normally, these do stripping of the comment section
907 # automatically.
908 #    RELEASE_CM:        Should be editted to reflect the release.
909 #    POST_PROCESS_O:    Post-processing for '.o' files.
910 #    POST_PROCESS_A:    Post-processing for '.a' files (currently null).
911 #    POST_PROCESS_SO:   Post-processing for '.so' files.
912 #    POST_PROCESS:      Post-processing for executable files (no suffix).
913 # Note that these macros are not completely generalized as they are to be
914 # used with the file name to be processed following.
915 #
916 # It is left as an exercise to Release Engineering to embellish the generation
917 # of the release comment string.
918 #
919 #       If this is a standard development build:
```

```
920 #               compress the comment section (mcs -c)
921 #               add the standard comment (mcs -a $(RELEASE_CM))
922 #               add the development specific comment (mcs -a $(DEV_CM))
923 #
924 #       If this is an installation build:
925 #               delete the comment section (mcs -d)
926 #               add the standard comment (mcs -a $(RELEASE_CM))
927 #               add the development specific comment (mcs -a $(DEV_CM))
928 #
929 #       If this is an release build:
930 #               delete the comment section (mcs -d)
931 #               add the standard comment (mcs -a $(RELEASE_CM))
932 #
933 # The following list of macros are used in the definition of RELEASE_CM
934 # which is used to label all binaries in the build:
935 #
936 #       RELEASE         Specific release of the build, eg: 5.2
937 #       RELEASE_MAJOR   Major version number part of $(RELEASE)
938 #       RELEASE_MINOR   Minor version number part of $(RELEASE)
939 #       VERSION         Version of the build (alpha, beta, Generic)
940 #       PATCHID         If this is a patch this value should contain
941 #                       the patchid value (eg: "Generic 100832-01"), otherwise
942 #                       it will be set to $(VERSION)
943 #       RELEASE_DATE    Date of the Release Build
944 #       PATCH_DATE      Date the patch was created, if this is blank it
945 #                       will default to the RELEASE_DATE
946 #
947 RELEASE_MAJOR=  5
948 RELEASE_MINOR=  11
949 RELEASE=        $(RELEASE_MAJOR).$(RELEASE_MINOR)
950 VERSION=        SunOS Development
951 PATCHID=        $(VERSION)
952 RELEASE_DATE=   release date not set
953 PATCH_DATE=     $(RELEASE_DATE)
954 RELEASE_CM=     "@($(POUND_SIGN))SunOS $(RELEASE) $(PATCHID) $(PATCH_DATE)"
955 DEV_CM=         "@($(POUND_SIGN))SunOS Internal Development: non-nightly build"

957 PROCESS_COMMENT=                   @?${MCS} -c -a $(RELEASE_CM) -a $(DEV_CM)
958 $(STRIP_COMMENTS)PROCESS_COMMENT=  @?${MCS} -d -a $(RELEASE_CM) -a $(DEV_CM)
959 $(RELEASE_BUILD)PROCESS_COMMENT=   @?${MCS} -d -a $(RELEASE_CM)

961 PROCESS_BUILDSTAMP=     @?${MCS} -n .ILLUMOS_buildstamp -d -a "'${BUILDSTAMP}'"

963 STRIP_STABS=                       :
964 $(RELEASE_BUILD)STRIP_STABS=       $(STRIP) -x $@

966 POST_PROCESS_O=         $(PROCESS_COMMENT) $@
967 POST_PROCESS_A=
968 POST_PROCESS_SO=        $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
969                         $(ELFSIGN_OBJECT)
970 POST_PROCESS=           $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
971                         $(ELFSIGN_OBJECT)

973 #
974 # chk4ubin is a tool that inspects a module for a symbol table
975 # ELF section size which can trigger an OBP bug on older platforms.
976 # This problem affects only specific sun4u bootable modules.
977 #
978 CHK4UBIN=       $(ONBLD_TOOLS)/bin/$(MACH)/chk4ubin
979 CHK4UBINFLAGS=
980 CHK4UBINARY=    $(CHK4UBIN) $(CHK4UBINFLAGS) $@

982 #
983 # PKGARCHIVE specifies the default location where packages should be
984 # placed if built.
985 #
```



```
986 $(RELEASE_BUILD)PKGARCHIVESUFFIX=       -nd
987 PKGARCHIVE=$(SRC)/../../packages/$(MACH)/nightly$(PKGARCHIVESUFFIX)

989 #
990 # The repositories will be created with these publisher settings.  To
991 # update an image to the resulting repositories, this must match the
992 # publisher name provided to "pkg set-publisher."
993 #
994 PKGPUBLISHER_REDIST=    on-nightly
995 PKGPUBLISHER_NONREDIST= on-extra

997 #       Default build rules which perform comment section post-processing.
998 #
999 .c:
1000        $(LINK.c) -o $@ $< $(LDLIBS)
1001        $(POST_PROCESS)
1002 .c.o:
1003        $(COMPILE.c) $(OUTPUT_OPTION) $< $(CTFCONVERT_HOOK)
1004        $(POST_PROCESS_O)
1005 .c.a:
1006        $(COMPILE.c) -o $% $<
1007        $(PROCESS_COMMENT) $%
1008        $(AR) $(ARFLAGS) $@ $%
1009        $(RM) $%
1010 .s.o:
1011        $(COMPILE.s) -o $@ $<
1012        $(POST_PROCESS_O)
1013 .s.a:
1014        $(COMPILE.s) -o $% $<
1015        $(PROCESS_COMMENT) $%
1016        $(AR) $(ARFLAGS) $@ $%
1017        $(RM) $%
1018 .cc:
1019        $(LINK.cc) -o $@ $< $(LDLIBS)
1020        $(POST_PROCESS)
1021 .cc.o:
1022        $(COMPILE.cc) $(OUTPUT_OPTION) $<
1023        $(POST_PROCESS_O)
1024 .cc.a:
1025        $(COMPILE.cc) -o $% $<
1026        $(AR) $(ARFLAGS) $@ $%
1027        $(PROCESS_COMMENT) $%
1028        $(RM) $%
1029 .y:
1030        $(YACC.y) $<
1031        $(LINK.c) -o $@ y.tab.c $(LDLIBS)
1032        $(POST_PROCESS)
1033        $(RM) y.tab.c
1034 .y.o:
1035        $(YACC.y) $<
1036        $(COMPILE.c) -o $@ y.tab.c $(CTFCONVERT_HOOK)
1037        $(POST_PROCESS_O)
1038        $(RM) y.tab.c
1039 .l:
1040        $(RM) $*.c
1041        $(LEX.l) $< > $*.c
1042        $(LINK.c) -o $@ $*.c -ll $(LDLIBS)
1043        $(POST_PROCESS)
1044        $(RM) $*.c
1045 .l.o:
1046        $(RM) $*.c
1047        $(LEX.l) $< > $*.c
1048        $(COMPILE.c) -o $@ $*.c $(CTFCONVERT_HOOK)
1049        $(POST_PROCESS_O)
1050        $(RM) $*.c
```

```
1052 .bin.o:
1053         $(COMPILE.b) -o $@ $<
1054         $(POST_PROCESS_O)

1056 .java.class:
1057         $(COMPILE.java) $<

1059 # Bourne and Korn shell script message catalog build rules.
1060 # We extract all gettext strings with sed(1) (being careful to permit
1061 # multiple gettext strings on the same line), weed out the dups, and
1062 # build the catalogue with awk(1).

1064 .sh.po .ksh.po:
1065         $(SED) -n -e ":a"                                   \
1066                   -e "h"                                        \
1067                   -e "s/.*gettext *\(\"[^\"]*\"\).*/\1/p"       \
1068                   -e "x"                                        \
1069                   -e "s/\(.*\)gettext *\"[^\"]*\"\(.*\)/\1\2/"  \
1070                   -e "t a"                                      \
1071                $< | sort -u | awk '{ print "msgid\t" $$0 "\nmsgstr" }' > $@

1073 #
1074 # Python and Perl executable and message catalog build rules.
1075 #
1076 .SUFFIXES: .pl .pm .py .pyc

1078 .pl:
1079         $(RM) $@;
1080         $(SED) -e "s@TEXT_DOMAIN@\"$(TEXT_DOMAIN)\"@" $< > $@;
1081         $(CHMOD) +x $@

1083 .py:
1084         $(RM) $@; $(CAT) $< > $@; $(CHMOD) +x $@

1086 .py.pyc:
1087         $(RM) $@
1088         $(PYTHON) -mpy_compile $<
1089         @[ $(<)c = $@ ] || $(MV) $(<)c $@

1091 .py.po:
1092         $(GNUXGETTEXT) $(GNUXGETFLAGS) -d $(<F:%.py=%) $< ;

1094 .pl.po .pm.po:
1095         $(XGETTEXT) $(XGETFLAGS) -d $(<F) $< ;
1096         $(RM)   $@ ;
1097         $(SED) "/^domain/d" < $(<F).po > $@ ;
1098         $(RM) $(<F).po

1100 #
1101 # When using xgettext, we want messages to go to the default domain,
1102 # rather than the specified one.  This special version of the
1103 # COMPILE.cpp macro effectively prevents expansion of TEXT_DOMAIN,
1104 # causing xgettext to put all messages into the default domain.
1105 #
1106 CPPFORPO=$(COMPILE.cpp:\"$(TEXT_DOMAIN)\"=TEXT_DOMAIN)

1108 .c.i:
1109         $(CPPFORPO) $< > $@

1111 .h.i:
1112         $(CPPFORPO) $< > $@

1114 .y.i:
1115         $(YACC) -d $<
1116         $(CPPFORPO) y.tab.c  > $@
1117         $(RM) y.tab.c
```



```
1119 .l.i:
1120         $(LEX) $<
1121         $(CPPFORPO) lex.yy.c  > $@
1122         $(RM) lex.yy.c

1124 .c.po:
1125         $(CPPFORPO) $< > $<.i
1126         $(BUILD.po)

1128 .y.po:
1129         $(YACC) -d $<
1130         $(CPPFORPO) y.tab.c  > $<.i
1131         $(BUILD.po)
1132         $(RM) y.tab.c

1134 .l.po:
1135         $(LEX) $<
1136         $(CPPFORPO) lex.yy.c  > $<.i
1137         $(BUILD.po)
1138         $(RM) lex.yy.c

1140 #
1141 # Rules to perform stylistic checks
1142 #
1143 .SUFFIXES: .x .xml .check .xmlchk

1145 .h.check:
1146         $(DOT_H_CHECK)

1148 .x.check:
1149         $(DOT_X_CHECK)

1151 .xml.xmlchk:
1152         $(MANIFEST_CHECK)

1154 #
1155 # Rules to process ONC+ Source partial files
1156 #
1157 %_onc_plus:     %
1158         @$(ECHO) "extracting code from $< ... "
1159         sed -n -e '/ONC_PLUS EXTRACT START/,/ONC_PLUS EXTRACT END/p' $<  > $@

1161 #
1162 # Include rules to render automated sccs get rules "safe".
1163 #
1164 include $(SRC)/Makefile.noget
```

```
**********************************************************
   13660 Mon Jun 25 17:12:20 2012
new/usr/src/pkg/manifests/developer-build-onbld.mf
include buildstamp in onbld
**********************************************************
```

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

  22 #
  23 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
  24 # Copyright 2010, Richard Lowe
  25 #

  27 set name=pkg.fmri value=pkg:/developer/build/onbld@$(PKGVERS)
  28 set name=pkg.description value="tools used to build the OS-Net consolidation"
  29 set name=pkg.summary value="OS-Net Build Tools"
  30 set name=info.classification \
  31     value="org.opensolaris.category.2008:Development/Distribution Tools"

  33 #
  34 # This package should not be incorporated.  This allows the tools
  35 # to be upgraded without upgrading the entire system.
  36 #
  37 set name=org.opensolaris.noincorp value=true
  38 set name=variant.arch value=$(ARCH)
  39 dir path=opt group=sys
  40 dir path=opt/onbld
  41 dir path=opt/onbld/bin
  42 dir path=opt/onbld/bin/$(ARCH)
  43 dir path=opt/onbld/env
  44 dir path=opt/onbld/etc
  45 dir path=opt/onbld/etc/exception_lists
  46 dir path=opt/onbld/gk
  47 dir path=opt/onbld/lib
  48 dir path=opt/onbld/lib/$(ARCH)
  49 dir path=opt/onbld/lib/perl
  50 dir path=opt/onbld/lib/python2.4
  51 dir path=opt/onbld/lib/python2.4/onbld
  52 dir path=opt/onbld/lib/python2.4/onbld/Checks
  53 dir path=opt/onbld/lib/python2.4/onbld/Scm
  54 dir path=opt/onbld/lib/python2.4/onbld/hgext
  55 dir path=opt/onbld/lib/python2.6
  56 dir path=opt/onbld/lib/python2.6/onbld
  57 dir path=opt/onbld/lib/python2.6/onbld/Checks
  58 dir path=opt/onbld/lib/python2.6/onbld/Scm
  59 dir path=opt/onbld/lib/python2.6/onbld/hgext
  60 dir path=opt/onbld/man
  61 dir path=opt/onbld/man/man1
```



```
  62 dir path=opt/onbld/man/sman1
  63 $(i386_ONLY)file path=opt/onbld/bin/$(ARCH)/aw mode=0555
  64 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/chk4ubin mode=0555
  65 file path=opt/onbld/bin/$(ARCH)/codereview mode=0555
  66 file path=opt/onbld/bin/$(ARCH)/cscope-fast mode=0555
  67 file path=opt/onbld/bin/$(ARCH)/ctfconvert mode=0555
  68 file path=opt/onbld/bin/$(ARCH)/ctfdump mode=0555
  69 file path=opt/onbld/bin/$(ARCH)/ctfmerge mode=0555
  70 file path=opt/onbld/bin/$(ARCH)/ctfstabs mode=0555
  71 file path=opt/onbld/bin/$(ARCH)/cw mode=0555
  72 $(i386_ONLY)file path=opt/onbld/bin/$(ARCH)/elfextract mode=0555
  73 file path=opt/onbld/bin/$(ARCH)/findunref mode=0555
  74 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/forth mode=0555
  75 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/forth_preload.so.1 mode=0555
  76 file path=opt/onbld/bin/$(ARCH)/install mode=0555
  77 file path=opt/onbld/bin/$(ARCH)/lintdump mode=0555
  78 $(i386_ONLY)file path=opt/onbld/bin/$(ARCH)/mbh_patch mode=0555
  79 file path=opt/onbld/bin/$(ARCH)/ndrgen mode=0555
  80 file path=opt/onbld/bin/$(ARCH)/ndrgen1 mode=0555
  81 file path=opt/onbld/bin/$(ARCH)/pmodes mode=0555
  82 file path=opt/onbld/bin/$(ARCH)/protocmp mode=0555
  83 file path=opt/onbld/bin/$(ARCH)/protolist mode=0555
  84 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/stabs mode=0555
  85 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/tokenize mode=0555
  86 $(sparc_ONLY)file path=opt/onbld/bin/$(ARCH)/tokenize.exe mode=0555
  87 file path=opt/onbld/bin/Install mode=0555
  88 file path=opt/onbld/bin/bindrop mode=0555
  89 file path=opt/onbld/bin/bldenv mode=0555
  90 file path=opt/onbld/bin/bringovercheck mode=0555
  91 file path=opt/onbld/bin/build_cscope mode=0555
  92 file path=opt/onbld/bin/buildstamp mode=0555
  93 file path=opt/onbld/bin/cddlchk mode=0555
  94 file path=opt/onbld/bin/check_rtime mode=0555
  95 file path=opt/onbld/bin/checkpaths mode=0555
  96 file path=opt/onbld/bin/checkproto mode=0555
  97 file path=opt/onbld/bin/copyrightchk mode=0555
  98 file path=opt/onbld/bin/cryptodrop mode=0555
  99 file path=opt/onbld/bin/cstyle mode=0555
 100 file path=opt/onbld/bin/ctfcvtptbl mode=0555
 101 file path=opt/onbld/bin/ctffindmod mode=0555
 102 file path=opt/onbld/bin/elfcmp mode=0555
 103 file path=opt/onbld/bin/elfsigncmp mode=0555
 104 file path=opt/onbld/bin/find_elf mode=0555
 105 file path=opt/onbld/bin/findcrypto mode=0555
 106 file path=opt/onbld/bin/flg.flp mode=0555
 107 file path=opt/onbld/bin/genoffsets mode=0555
 108 file path=opt/onbld/bin/get_depend_info mode=0555
 109 file path=opt/onbld/bin/git-pbchk mode=0555
 110 file path=opt/onbld/bin/hdrchk mode=0555
 111 file path=opt/onbld/bin/hg-active mode=0555
 112 file path=opt/onbld/bin/hgsetup mode=0555
 113 file path=opt/onbld/bin/interface_check mode=0555
 114 file path=opt/onbld/bin/interface_cmp mode=0555
 115 file path=opt/onbld/bin/jstyle mode=0555
 116 file path=opt/onbld/bin/make_pkg_db mode=0555
 117 file path=opt/onbld/bin/mapfilechk mode=0555
 118 file path=opt/onbld/bin/mkreadme_osol mode=0555
 119 file path=opt/onbld/bin/mktpl mode=0555
 120 file path=opt/onbld/bin/nightly mode=0555
 121 file path=opt/onbld/bin/onu mode=0555
 122 file path=opt/onbld/bin/protocmp.terse mode=0555
 123 file path=opt/onbld/bin/sccscheck mode=0555
 124 file path=opt/onbld/bin/signit mode=0555
 125 file path=opt/onbld/bin/signproto mode=0555
 126 file path=opt/onbld/bin/validate_flg mode=0555
 127 file path=opt/onbld/bin/validate_paths mode=0555
```

```
128 file path=opt/onbld/bin/validate_pkg mode=0555
129 file path=opt/onbld/bin/wdiff mode=0555
130 file path=opt/onbld/bin/webrev mode=0555
131 file path=opt/onbld/bin/which_scm mode=0555
132 file path=opt/onbld/bin/ws mode=0555
133 file path=opt/onbld/bin/wsdiff mode=0555
134 file path=opt/onbld/bin/xref mode=0555
135 file path=opt/onbld/bin/xref.mk
136 file path=opt/onbld/env/developer
137 file path=opt/onbld/env/gatekeeper
138 file path=opt/onbld/env/illumos
139 file path=opt/onbld/etc/SampleLinks
140 file path=opt/onbld/etc/SamplePkgLinks
141 file path=opt/onbld/etc/exception_lists/check_rtime
142 file path=opt/onbld/etc/exception_lists/interface_check
143 file path=opt/onbld/etc/exception_lists/interface_cmp
144 file path=opt/onbld/etc/hgstyle
145 file path=opt/onbld/etc/its.conf
146 file path=opt/onbld/etc/its.reg
147 file path=opt/onbld/gk/.cshrc
148 file path=opt/onbld/gk/.login
149 file path=opt/onbld/gk/gen_make.machines mode=0755
150 file path=opt/onbld/lib/$(ARCH)/libdwarf.so.1
151 file path=opt/onbld/lib/perl/onbld_elfmod.pm
152 file path=opt/onbld/lib/perl/onbld_elfmod_vertype.pm
153 file path=opt/onbld/lib/python2.4/onbld/Checks/CStyle.py mode=0444
154 file path=opt/onbld/lib/python2.4/onbld/Checks/CStyle.pyc mode=0444
155 file path=opt/onbld/lib/python2.4/onbld/Checks/Cddl.py mode=0444
156 file path=opt/onbld/lib/python2.4/onbld/Checks/Cddl.pyc mode=0444
157 file path=opt/onbld/lib/python2.4/onbld/Checks/CmtBlk.py mode=0444
158 file path=opt/onbld/lib/python2.4/onbld/Checks/CmtBlk.pyc mode=0444
159 file path=opt/onbld/lib/python2.4/onbld/Checks/Comments.py mode=0444
160 file path=opt/onbld/lib/python2.4/onbld/Checks/Comments.pyc mode=0444
161 file path=opt/onbld/lib/python2.4/onbld/Checks/Copyright.py mode=0444
162 file path=opt/onbld/lib/python2.4/onbld/Checks/Copyright.pyc mode=0444
163 file path=opt/onbld/lib/python2.4/onbld/Checks/DbLookups.py mode=0444
164 file path=opt/onbld/lib/python2.4/onbld/Checks/DbLookups.pyc mode=0444
165 file path=opt/onbld/lib/python2.4/onbld/Checks/HdrChk.py mode=0444
166 file path=opt/onbld/lib/python2.4/onbld/Checks/HdrChk.pyc mode=0444
167 file path=opt/onbld/lib/python2.4/onbld/Checks/JStyle.py mode=0444
168 file path=opt/onbld/lib/python2.4/onbld/Checks/JStyle.pyc mode=0444
169 file path=opt/onbld/lib/python2.4/onbld/Checks/Keywords.py mode=0444
170 file path=opt/onbld/lib/python2.4/onbld/Checks/Keywords.pyc mode=0444
171 file path=opt/onbld/lib/python2.4/onbld/Checks/Mapfile.py mode=0444
172 file path=opt/onbld/lib/python2.4/onbld/Checks/Mapfile.pyc mode=0444
173 file path=opt/onbld/lib/python2.4/onbld/Checks/ProcessCheck.py mode=0444
174 file path=opt/onbld/lib/python2.4/onbld/Checks/ProcessCheck.pyc mode=0444
175 file path=opt/onbld/lib/python2.4/onbld/Checks/__init__.py mode=0444
176 file path=opt/onbld/lib/python2.4/onbld/Checks/__init__.pyc mode=0444
177 file path=opt/onbld/lib/python2.4/onbld/Scm/Backup.py mode=0444
178 file path=opt/onbld/lib/python2.4/onbld/Scm/Backup.pyc mode=0444
179 file path=opt/onbld/lib/python2.4/onbld/Scm/Version.py mode=0444
180 file path=opt/onbld/lib/python2.4/onbld/Scm/Version.pyc mode=0444
181 file path=opt/onbld/lib/python2.4/onbld/Scm/WorkSpace.py mode=0444
182 file path=opt/onbld/lib/python2.4/onbld/Scm/WorkSpace.pyc mode=0444
183 file path=opt/onbld/lib/python2.4/onbld/Scm/__init__.py mode=0444
184 file path=opt/onbld/lib/python2.4/onbld/Scm/__init__.pyc mode=0444
185 file path=opt/onbld/lib/python2.4/onbld/__init__.py mode=0444
186 file path=opt/onbld/lib/python2.4/onbld/__init__.pyc mode=0444
187 file path=opt/onbld/lib/python2.4/onbld/hgext/__init__.py mode=0444
188 file path=opt/onbld/lib/python2.4/onbld/hgext/__init__.pyc mode=0444
189 file path=opt/onbld/lib/python2.4/onbld/hgext/cdm.py mode=0444
190 file path=opt/onbld/lib/python2.6/onbld/Checks/CStyle.py mode=0444
191 file path=opt/onbld/lib/python2.6/onbld/Checks/CStyle.pyc mode=0444
192 file path=opt/onbld/lib/python2.6/onbld/Checks/Cddl.py mode=0444
193 file path=opt/onbld/lib/python2.6/onbld/Checks/Cddl.pyc mode=0444
```



```
194 file path=opt/onbld/lib/python2.6/onbld/Checks/CmtBlk.py mode=0444
195 file path=opt/onbld/lib/python2.6/onbld/Checks/CmtBlk.pyc mode=0444
196 file path=opt/onbld/lib/python2.6/onbld/Checks/Comments.py mode=0444
197 file path=opt/onbld/lib/python2.6/onbld/Checks/Comments.pyc mode=0444
198 file path=opt/onbld/lib/python2.6/onbld/Checks/Copyright.py mode=0444
199 file path=opt/onbld/lib/python2.6/onbld/Checks/Copyright.pyc mode=0444
200 file path=opt/onbld/lib/python2.6/onbld/Checks/DbLookups.py mode=0444
201 file path=opt/onbld/lib/python2.6/onbld/Checks/DbLookups.pyc mode=0444
202 file path=opt/onbld/lib/python2.6/onbld/Checks/HdrChk.py mode=0444
203 file path=opt/onbld/lib/python2.6/onbld/Checks/HdrChk.pyc mode=0444
204 file path=opt/onbld/lib/python2.6/onbld/Checks/JStyle.py mode=0444
205 file path=opt/onbld/lib/python2.6/onbld/Checks/JStyle.pyc mode=0444
206 file path=opt/onbld/lib/python2.6/onbld/Checks/Keywords.py mode=0444
207 file path=opt/onbld/lib/python2.6/onbld/Checks/Keywords.pyc mode=0444
208 file path=opt/onbld/lib/python2.6/onbld/Checks/Mapfile.py mode=0444
209 file path=opt/onbld/lib/python2.6/onbld/Checks/Mapfile.pyc mode=0444
210 file path=opt/onbld/lib/python2.6/onbld/Checks/ProcessCheck.py mode=0444
211 file path=opt/onbld/lib/python2.6/onbld/Checks/ProcessCheck.pyc mode=0444
212 file path=opt/onbld/lib/python2.6/onbld/Checks/__init__.py mode=0444
213 file path=opt/onbld/lib/python2.6/onbld/Checks/__init__.pyc mode=0444
214 file path=opt/onbld/lib/python2.6/onbld/Scm/Backup.py mode=0444
215 file path=opt/onbld/lib/python2.6/onbld/Scm/Backup.pyc mode=0444
216 file path=opt/onbld/lib/python2.6/onbld/Scm/Version.py mode=0444
217 file path=opt/onbld/lib/python2.6/onbld/Scm/Version.pyc mode=0444
218 file path=opt/onbld/lib/python2.6/onbld/Scm/WorkSpace.py mode=0444
219 file path=opt/onbld/lib/python2.6/onbld/Scm/WorkSpace.pyc mode=0444
220 file path=opt/onbld/lib/python2.6/onbld/Scm/__init__.py mode=0444
221 file path=opt/onbld/lib/python2.6/onbld/Scm/__init__.pyc mode=0444
222 file path=opt/onbld/lib/python2.6/onbld/__init__.py mode=0444
223 file path=opt/onbld/lib/python2.6/onbld/__init__.pyc mode=0444
224 file path=opt/onbld/lib/python2.6/onbld/hgext/__init__.py mode=0444
225 file path=opt/onbld/lib/python2.6/onbld/hgext/__init__.pyc mode=0444
226 file path=opt/onbld/lib/python2.6/onbld/hgext/cdm.py mode=0444
227 file path=opt/onbld/man/man1/Install.1
228 file path=opt/onbld/man/man1/bldenv.1
229 file path=opt/onbld/man/man1/bringovercheck.1
230 file path=opt/onbld/man/man1/cddlchk.1
231 file path=opt/onbld/man/man1/check_rtime.1
232 file path=opt/onbld/man/man1/checkpaths.1
233 file path=opt/onbld/man/man1/codereview.1
234 file path=opt/onbld/man/man1/cstyle.1
235 file path=opt/onbld/man/man1/cw.1
236 file path=opt/onbld/man/man1/find_elf.1
237 file path=opt/onbld/man/man1/findunref.1
238 file path=opt/onbld/man/man1/flg.flp.1
239 file path=opt/onbld/man/man1/git-pbchk.1
240 file path=opt/onbld/man/man1/hdrchk.1
241 file path=opt/onbld/man/man1/hgsetup.1
242 file path=opt/onbld/man/man1/interface_check.1
243 file path=opt/onbld/man/man1/interface_cmp.1
244 file path=opt/onbld/man/man1/jstyle.1
245 file path=opt/onbld/man/man1/lintdump.1
246 file path=opt/onbld/man/man1/mapfilechk.1
247 file path=opt/onbld/man/man1/ndrgen.1
248 file path=opt/onbld/man/man1/nightly.1
249 file path=opt/onbld/man/man1/onu.1
250 file path=opt/onbld/man/man1/sccscheck.1
251 file path=opt/onbld/man/man1/signit.1
252 file path=opt/onbld/man/man1/signproto.1
253 file path=opt/onbld/man/man1/webrev.1
254 file path=opt/onbld/man/man1/which_scm.1
255 file path=opt/onbld/man/man1/ws.1
256 file path=opt/onbld/man/man1/wsdiff.1
257 file path=opt/onbld/man/man1/xref.1
258 file path=opt/onbld/man/sman1/get_depend_info.1
259 file path=opt/onbld/man/sman1/make_pkg_db.1
```

```
260 hardlink path=opt/onbld/bin/$(ARCH)/install.bin target=./install
261 legacy pkg=SUNWonbld desc="tools used to build the OS-Net consolidation" \
262     name="OS-Net Build Tools" version=11.11,REV=2009.10.22
263 license cr_Sun license=cr_Sun
264 license lic_CDDL license=lic_CDDL
265 license usr/src/tools/ctf/dwarf/THIRDPARTYLICENSE \
266     license=usr/src/tools/ctf/dwarf/THIRDPARTYLICENSE
267 license usr/src/tools/onbld/THIRDPARTYLICENSE \
268     license=usr/src/tools/onbld/THIRDPARTYLICENSE
269 link path=opt/onbld/bin/git-nits target=git-pbchk
270 link path=opt/onbld/lib/python target=python2.4
271 link path=opt/onbld/man/man1/git-nits.1 target=git-pbchk.1
272 # DbLookups.py requires elementtree
273 depend fmri=library/python-2/python-extra-24 type=require
274 # webrev(1) requires ps2pdf
275 depend fmri=print/filter/ghostscript type=require
276 # hgsetup(1) uses check-hostname(1) and nightly sendmail(1M)
277 depend fmri=service/network/smtp/sendmail type=require
278 # nightly(1) uses wget
279 depend fmri=web/wget type=require
```

```
**********************************************************
    3649 Mon Jun 25 17:12:20 2012
new/usr/src/tools/scripts/Makefile
include build stamp in unix
**********************************************************
```

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

SHELL=/usr/bin/ksh93

SHFILES= \
        Install \
        bindrop \
        bldenv \
        build_cscope \
        bringovercheck \
        checkpaths \
        checkproto \
        cryptodrop \
        cstyle \
        elfcmp \
        flg.flp \
        genoffsets \
        hgsetup \
        nightly \
        onu \
        protocmp.terse \
        sccscheck \
        webrev \
        which_scm \
        ws \
        xref \
        buildstamp
        xref

PERLFILES= \
        check_rtime \
        find_elf \
        interface_check \
        interface_cmp \
        jstyle \
        mkreadme_osol \
        mktpl \
        validate_flg \
```



```
        validate_paths \
        wdiff

PERLMODULES= \
        onbld_elfmod.pm \
        onbld_elfmod_vertype.pm


PYFILES= \
        cddlchk \
        copyrightchk \
        git-pbchk \
        hdrchk \
        hg-active \
        mapfilechk \
        validate_pkg \
        wsdiff

SCRIPTLINKS= \
        git-nits

MAN1FILES= \
        Install.1 \
        bldenv.1 \
        bringovercheck.1 \
        cddlchk.1 \
        checkpaths.1 \
        check_rtime.1 \
        cstyle.1 \
        find_elf.1 \
        flg.flp.1 \
        git-pbchk.1 \
        hdrchk.1 \
        interface_check.1 \
        interface_cmp.1 \
        hgsetup.1 \
        jstyle.1 \
        mapfilechk.1 \
        nightly.1 \
        onu.1 \
        sccscheck.1 \
        webrev.1 \
        which_scm.1 \
        ws.1 \
        wsdiff.1 \
        xref.1

MAN1LINKS= \
        git-nits.1

MAKEFILES= \
        xref.mk

ETCFILES= \
        hgstyle \
        its.conf \
        its.reg

EXCEPTFILES= \
        check_rtime \
        interface_check \
        interface_cmp

CLEANFILES = $(SHFILES) $(PERLFILES) $(PYFILES) bldenv.1

include ../Makefile.tools
```

```
128 ROOTONBLDSCRIPTLINKS = $(SCRIPTLINKS:%=$(ROOTONBLDBIN)/%)
129 ROOTONBLDMAN1LINKS = $(MAN1LINKS:%=$(ROOTONBLDMAN1)/%)

131 $(ROOTONBLDETCFILES)    := FILEMODE=    644
132 $(ROOTONBLDEXCEPTFILES) := FILEMODE=    644
133 $(ROOTONBLDPERLMODULES) := FILEMODE=    644
134 $(ROOTONBLDMAKEFILES)   := FILEMODE=    644
135 $(ROOTONBLDMAN1FILES)   := FILEMODE=    644

137 .KEEP_STATE:

139 all:    $(SHFILES) $(PERLFILES) $(PERLMODULES) $(PYFILES) \
140         $(MAN1FILES) $(MAKEFILES)

142 $(ROOTONBLDBIN)/git-nits:
143         $(RM) $(ROOTONBLDBIN)/git-nits
144         $(SYMLINK) git-pbchk $(ROOTONBLDBIN)/git-nits

146 $(ROOTONBLDMAN1)/git-nits.1:
147         $(RM) $(ROOTONBLDMAN1)/git-nits.1
148         $(SYMLINK) git-pbchk.1 $(ROOTONBLDMAN1)/git-nits.1

150 install: all .WAIT $(ROOTONBLDSHFILES) $(ROOTONBLDPERLFILES)    \
151                 $(ROOTONBLDPERLMODULES) $(ROOTONBLDPYFILES)     \
152                 $(ROOTONBLDSCRIPTLINKS) $(ROOTONBLDMAN1FILES)   \
153                 $(ROOTONBLDMAKEFILES) $(ROOTONBLDETCFILES)      \
154                 $(ROOTONBLDEXCEPTFILES) $(ROOTONBLDMAN1LINKS)

156 clean:
157         $(RM) $(CLEANFILES)

159 bldenv: bldenv.sh stdenv.sh
160         $(RM) "$@"
161         sed -e '/# STDENV_START/ r stdenv.sh' bldenv.sh > "$@"
162         # Check for shell lint and fail if we hit warnings
163         shlintout="$$( /usr/bin/ksh93 -n "$@" 2>&1 )" ; \
164                 [[ "$${shlintout}" != "" ]] && \
165                 { print -r -- "$${shlintout}" ; false ; } || true
166         $(CHMOD) +x "$@"

168 bldenv.1: bldenv
169         $(RM) "$@"
170         (set +o errexit ; ksh93 $? --nroff ; true) 2>&1 | \
171         sed 's/\.DS/.nf/g;s/\.DE/.fi/' > "$@"

173 nightly: nightly.sh stdenv.sh
174         $(RM) "$@"
175         sed -e '/# STDENV_START/ r stdenv.sh' nightly.sh > nightly
176         $(CHMOD) +x "$@"

178 include ../Makefile.targ
```

```
**********************************************************
    1474 Mon Jun 25 17:12:20 2012
new/usr/src/tools/scripts/buildstamp.sh
include build stamp in unix
**********************************************************
```

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

#
# Copyright 2012 Joshua M. Clulow <josh@sysmgr.org>
#

WHICH_SCM=which_scm
CW=cw
NAWK=nawk


echo "date: $(date)"
echo "uname: $(uname -a)"

$WHICH_SCM | read scm_type junk || exit 1
cmd=
if [[ $scm_type == "git" ]]; then
        cmd="git rev-parse --verify HEAD"
elif [[ $scm_type == "mercurial" ]]; then
        cmd="hg log -r tip --template {node}"
fi
if [[ -n $cmd ]]; then
        echo "scm: ${scm_type} $($cmd)"
fi

$CW -_versions 2>&1 | $NAWK '
  /^primary:/ { print; watch = 1; next; }
  watch == 1  { print("primaryversion: " $0); watch = 0; next; }
  /^shadow:/  { print; watch = 2; next; }
  watch == 2  { print("shadowversion: " $0); watch = 0; next; }
'

exit 0
```

```
**********************************************************
   14150 Mon Jun 25 17:12:21 2012
new/usr/src/uts/Makefile.targ
include build stamp in unix
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
#
#       This Makefiles contains the common targets and definitions for
#       all kernels. It is to be included in the Makefiles for specific
#       implementation architectures and processor architecture dependent
#       modules: i.e.: all driving kernel Makefiles.
#


#
#       Default rule for building the lint library directory:
#
$(LINT_LIB_DIR):
        -@mkdir -p $@ 2> /dev/null

#
#       All C objects depend on inline files. However, cc(1) doesn't generate
#       the correct dependency info. Also, these Makefiles don't contain a
#       separate list of C-derived object files (but it is light weight to
#       let the assembler files think they depend upon this when they don't).
#       Fortunately, the inline files won't change very often. So, for now,
#       all objects depend on the inline files. Remove this when the inliner
#       is fixed to drop correct dependency information.
#
$(OBJECTS): $(INLINES)

#
#       Partially link .o files to generate the kmod. The fake dependency
#       on modstubs simplifies things...
#       ELFSIGN_MOD is defined in the individual KCF plug-in modules Makefiles,
#       and will sign the ELF objects using elfsign(1).
#
$(BINARY):              $(OBJECTS)
        $(LD) -r $(LDFLAGS) -o $@ $(OBJECTS)
        $(CTFMERGE_UNIQUIFY_AGAINST_GENUNIX)
        $(POST_PROCESS)
        $(ELFSIGN_MOD)

#
#       This target checks each kmod for undefined entry points. It does not
#       modify the kmod in any way.
```



```
#
$(MODULE).check:        FRC
        @BUILD_TYPE=DBG32 $(MAKE) $(MODULE).check.targ

$(MODULE).check.targ:   $(BINARY) $(OBJECTS) $(EXTRA_CHECK_OBJS) $(UNIX_O) $(MOD
        $(LD) -o /dev/null $(OBJECTS) $(EXTRA_CHECK_OBJS) $(UNIX_O) $(MODSTUBS_O

#
#       Module lint library construction targets.
#
MOD_LINT_LIB    = $(LINT_LIB_DIR)/llib-l$(LINT_MODULE).ln

$(MOD_LINT_LIB):        $(LINT_LIB_DIR) $(LINTS)
        @-$(ECHO) "\n$(OBJS_DIR)/$(MODULE): (library construction):"
        @($(LINT) -o $(LINT_MODULE)-$(OBJS_DIR) \
                $(LINTFLAGS) $(LINTS) $(LTAIL))
        @$(MV) llib-l$(LINT_MODULE)-$(OBJS_DIR).ln $@

$(LINT_MODULE).lint:    $(MOD_LINT_LIB) $(LINT_LIB) $(GEN_LINT_LIB)
        @-$(ECHO) "\n$(OBJS_DIR)/$(LINT_MODULE): global crosschecks:"
        @($(LINT) $(LINTFLAGS) $(MOD_LINT_LIB) \
                $(LINT_LIB) $(GEN_LINT_LIB) $(LTAIL))

#
# Since assym.h is a derived file, the dependency must be explicit for
# all files including this file. (This is only actually required in the
# instance when the .nse_depinfo file does not exist.) It may seem that
# the lint targets should also have a similar dependency, but they don't
# since only C headers are included when #defined(lint) is true. The
# actual lists are defined in */Makefile.files.
#
$(ASSYM_DEPS:%=$(OBJS_DIR)/%):  $(DSF_DIR)/$(OBJS_DIR)/assym.h

#
#       Everybody need to know how to create a modstubs.o built with the
#       appropriate flags and located in the appropriate location.
#
$(MODSTUBS_O):  $(MODSTUBS)
        $(COMPILE.s) -o $@ $(MODSTUBS)

$(LINTS_DIR)/modstubs.ln:       $(MODSTUBS)
        @($(LHEAD) $(LINT.s) $(MODSTUBS) $(LTAIL))

#
# Build the source file which contains the kernel's utsname,
# with release, version and machine set as follows:
#
#       release: contents of $(RELEASE) (Spaces replaced by '_')
#       version: contents of $(PATCHID) (Spaces replaced by '_')
#       machine: contents of $(UNAME_M)
#
# Build environment information is only contained in the comment section.
#
# The version string, normally the variable VERSION, is set to display
# environmental information temporarily while in development because
# it provides a little more useful information.
#
VERSION_STRING  = ($(ECHO) $$LOGNAME [\`basename $$CODEMGR_WS\`] \\\c; date +%D)
$(INTERNAL_RELEASE_BUILD)VERSION_STRING = $(ECHO) $(PATCHID)

$(OBJS_DIR)/vers.o: $(OBJECTS)
        $(COMPILE.c) -DUTS_RELEASE=\"'$(ECHO) $(RELEASE) | sed -e 's/ /_/g''\" \
            -DUTS_VERSION=\"'$(VERSION_STRING) | sed -e 's/ /_/g''\" \
            -DUTS_PLATFORM=\"$(UNAME_M)\" -o $@ $(SRC)/uts/common/os/vers.c
        $(CTFCONVERT_O)
        $(PROCESS_BUILDSTAMP) $@
```

```
128         $(POST_PROCESS_O)

130 $(LINTS_DIR)/vers.ln: $(SRC)/uts/common/os/vers.c
131         @($(LHEAD) $(LINT.c) -DUTS_RELEASE=\"\" -DUTS_VERSION=\"\" \
132             -DUTS_PLATFORM=\"\" $(SRC)/uts/common/os/vers.c $(LTAIL))

134 #
135 #       Installation targets and rules:
136 #
137 $(ROOT_MOD_DIR) $(USR_MOD_DIR):
138         -$(INS.dir)

140 $(ROOT_MOD_DIRS_32):    $(ROOT_MOD_DIR)
141         -$(INS.dir)

143 $(USR_MOD_DIRS_32):     $(USR_MOD_DIR)
144         -$(INS.dir)

146 $(ROOT_MOD_DIR)/%:      $(OBJS_DIR)/% $(ROOT_MOD_DIR) FRC
147         $(INS.file)

149 $(ROOT_CPU_DIR)/%:      $(OBJS_DIR)/% $(ROOT_CPU_DIR) FRC
150         $(INS.file)

152 $(ROOT_DRV_DIR)/%:      $(OBJS_DIR)/% $(ROOT_DRV_DIR) FRC
153         $(INS.file)

155 $(ROOT_DTRACE_DIR)/%:   $(OBJS_DIR)/% $(ROOT_DTRACE_DIR) FRC
156         $(INS.file)

158 $(ROOT_EXEC_DIR)/%:     $(OBJS_DIR)/% $(ROOT_EXEC_DIR) FRC
159         $(INS.file)

161 $(ROOT_FS_DIR)/%:       $(OBJS_DIR)/% $(ROOT_FS_DIR) FRC
162         $(INS.file)

164 $(ROOT_SCHED_DIR)/%:    $(OBJS_DIR)/% $(ROOT_SCHED_DIR) FRC
165         $(INS.file)

167 $(ROOT_SOCK_DIR)/%:     $(OBJS_DIR)/% $(ROOT_SOCK_DIR) FRC
168         $(INS.file)

170 $(ROOT_STRMOD_DIR)/%:   $(OBJS_DIR)/% $(ROOT_STRMOD_DIR) FRC
171         $(INS.file)

173 $(ROOT_IPP_DIR)/%:      $(OBJS_DIR)/% $(ROOT_IPP_DIR) FRC
174         $(INS.file)

176 $(ROOT_SYS_DIR)/%:      $(OBJS_DIR)/% $(ROOT_SYS_DIR) FRC
177         $(INS.file)

179 $(ROOT_MISC_DIR)/%:     $(OBJS_DIR)/% $(ROOT_MISC_DIR) FRC
180         $(INS.file)

182 $(ROOT_DACF_DIR)/%:     $(OBJS_DIR)/% $(ROOT_DACF_DIR) FRC
183         $(INS.file)

185 $(ROOT_BRAND_DIR)/%:    $(OBJS_DIR)/% $(ROOT_BRAND_DIR) FRC
186         $(INS.file)

188 $(ROOT_CRYPTO_DIR)/%:   $(OBJS_DIR)/% $(ROOT_CRYPTO_DIR) FRC
189         $(INS.file)

191 $(ROOT_KGSS_DIR)/%:     $(OBJS_DIR)/% $(ROOT_KGSS_DIR) FRC
192         $(INS.file)
```



```
194 $(ROOT_SCSI_VHCI_DIR)/%: $(OBJS_DIR)/% $(ROOT_SCSI_VHCI_DIR) FRC
195         $(INS.file)

197 $(ROOT_PMCS_FW_DIR)/%:  $(OBJS_DIR)/% $(ROOT_PMCS_FW_DIR) FRC
198         $(INS.file)

200 $(ROOT_QLC_FW_DIR)/%:   $(OBJS_DIR)/% $(ROOT_QLC_FW_DIR) FRC
201         $(INS.file)

203 $(ROOT_EMLXS_FW_DIR)/%: $(OBJS_DIR)/% $(ROOT_EMLXS_FW_DIR) FRC
204         $(INS.file)

206 $(ROOT_MACH_DIR)/%:     $(OBJS_DIR)/% $(ROOT_MACH_DIR) FRC
207         $(INS.file)

209 $(ROOT_FONT_DIR)/%:     $(OBJS_DIR)/% $(ROOT_MOD_DIR) $(ROOT_FONT_DIR) FRC
210         $(INS.file)

212 $(ROOT_MAC_DIR)/%:      $(OBJS_DIR)/% $(ROOT_MOD_DIR) $(ROOT_MAC_DIR) FRC
213         $(INS.file)

215 $(USR_DRV_DIR)/%:       $(OBJS_DIR)/% $(USR_DRV_DIR) FRC
216         $(INS.file)

218 $(USR_EXEC_DIR)/%:      $(OBJS_DIR)/% $(USR_EXEC_DIR) FRC
219         $(INS.file)

221 $(USR_FS_DIR)/%:        $(OBJS_DIR)/% $(USR_FS_DIR) FRC
222         $(INS.file)

224 $(USR_SCHED_DIR)/%:     $(OBJS_DIR)/% $(USR_SCHED_DIR) FRC
225         $(INS.file)

227 $(USR_SOCK_DIR)/%:      $(OBJS_DIR)/% $(USR_SOCK_DIR) FRC
228         $(INS.file)

230 $(USR_STRMOD_DIR)/%:    $(OBJS_DIR)/% $(USR_STRMOD_DIR) FRC
231         $(INS.file)

233 $(USR_SYS_DIR)/%:       $(OBJS_DIR)/% $(USR_SYS_DIR) FRC
234         $(INS.file)

236 $(USR_MISC_DIR)/%:      $(OBJS_DIR)/% $(USR_MISC_DIR) FRC
237         $(INS.file)

239 $(USR_DACF_DIR)/%:      $(OBJS_DIR)/% $(USR_DACF_DIR) FRC
240         $(INS.file)

242 $(USR_PCBE_DIR)/%:      $(OBJS_DIR)/% $(USR_PCBE_DIR) FRC
243         $(INS.file)

245 $(USR_DTRACE_DIR)/%:    $(OBJS_DIR)/% $(USR_DTRACE_DIR) FRC
246         $(INS.file)

248 $(USR_BRAND_DIR)/%:     $(OBJS_DIR)/% $(USR_BRAND_DIR) FRC
249         $(INS.file)

251 $(ROOT_KICONV_DIR)/%:   $(OBJS_DIR)/% $(ROOT_KICONV_DIR) FRC
252         $(INS.file)

254 include $(SRC)/Makefile.psm.targ

256 #
257 #       Target for 64b modules
258 #
259 $(ROOT_KERN_DIR_64):
```

```
260         -$(INS.dir)

262 $(ROOT_KERN_DIR_64)/%:  $(OBJS_DIR)/% $(ROOT_KERN_DIR_64) FRC
263         $(INS.file)

265 %/$(SUBDIR64):          %
266         -$(INS.dir)

268 #
269 #       Targets for '.conf' file installation.
270 #
271 $(ROOT_CONFFILE):       $(SRC_CONFFILE) $(ROOT_CONFFILE:%/$(CONFFILE)=%)
272         $(INS.conffile)

274 #
275 #       Targets for creating links between common platforms. ROOT_PLAT_LINKS
276 #       are are the /platform level while ROOT_PLAT_LINKS_2 are one level
277 #       down (/platform/`uname -i`/{lib|sbin|kernel}.
278 #
279 $(ROOT_PLAT_LINKS):
280         $(INS.slink1)

282 $(ROOT_PLAT_LINKS_2):
283         $(INS.slink2)

285 $(USR_PLAT_LINKS):
286         $(INS.slink1)

288 $(USR_PLAT_LINKS_2):
289         $(INS.slink2)

291 #
292 # multiple builds support
293 #
294 def $(DEF_DEPS)                 := TARGET = def
295 all $(ALL_DEPS)                 := TARGET = all
296 clean $(CLEAN_DEPS)             := TARGET = clean
297 clobber $(CLOBBER_DEPS)         := TARGET = clobber
298 lint $(LINT_DEPS)               := TARGET = lint
299 modlintlib $(MODLINTLIB_DEPS)   := TARGET = modlintlib
300 modlist $(MODLIST_DEPS)         := TARGET = modlist
301 modlist $(MODLIST_DEPS)         := NO_STATE= -K $$MODSTATE$$$$
302 clean.lint $(CLEAN_LINT_DEPS)   := TARGET = clean.lint
303 install $(INSTALL_DEPS)         := TARGET = install
304 symcheck $(SYM_DEPS)            := TARGET = symcheck

306 ALL_TARGS       = def all clean clobber lint modlintlib \
307                   clean.lint lintlib install symcheck

309 ALL_OBJ32       = $(ALL_TARGS:%=%.obj32)

311 $(ALL_OBJ32):   FRC
312         @BUILD_TYPE=OBJ32 VERSION='$(VERSION)' $(MAKE) $(NO_STATE) $(TARGET).tar

314 ALL_DEBUG32     = $(ALL_TARGS:%=%.debug32)

316 $(ALL_DEBUG32): FRC
317         @BUILD_TYPE=DBG32 VERSION='$(VERSION)' $(MAKE) $(NO_STATE) $(TARGET).tar

319 ALL_OBJ64       = $(ALL_TARGS:%=%.obj64)

321 $(ALL_OBJ64):   FRC
322         @BUILD_TYPE=OBJ64 VERSION='$(VERSION)' $(MAKE) $(NO_STATE) $(TARGET).tar

324 ALL_DEBUG64     = $(ALL_TARGS:%=%.debug64)
```



```
326 $(ALL_DEBUG64): FRC
327         @BUILD_TYPE=DBG64 VERSION='$(VERSION)' $(MAKE) $(NO_STATE) $(TARGET).tar

329 #
330 #       Currently only the IP module needs symbol checking on obj64.
331 #       Other modules have the same global-objs nm output for debug64 and obj64.
332 #
333 $(SISCHECK_DEPS):       $(DEF_DEPS)
334         @TARG='$(ECHO) $@ | $(CUT) -d'.' -f2'; \
335         MODSYMS=$(MODULE).symbols.$$TARG; \
336         if [ -f "$(MODULE).global-objs.$$TARG" ]; then \
337                 $(GREP) -v '#' $(MODULE).global-objs.$$TARG |$(GREP) . | \
338                     $(SORT) -u > $$MODSYMS.tmp; \
339                 $(NM) $$TARG/$(MODULE) |$(GREP) OBJT |$(GREP) -v UNDEF | \
340                     $(CUT) -d'|' -f8 |$(GREP) -v '^___const_' | \
341                     $(GREP) -v '\.[0-9]*$$' |$(SORT) -u \
342                     > $$MODSYMS.tmp.new; \
343                 $(DIFF) $$MODSYMS.tmp $$MODSYMS.tmp.new > $$MODSYMS.diff || \
344                     ($(ECHO) "warning: $(MODULE) symbol checking:" \
345                     "global variable(s) introduced and/or removed."; \
346                     $(CAT) $$MODSYMS.diff; exit 1) \
347         fi

349 $(SISCLEAN_DEPS):
350         -TARG='$(ECHO) $@ | $(CUT) -d'.' -f2'; \
351         MODSYMS=$(MODULE).symbols.$$TARG; \
352         $(RM) $$MODSYMS.tmp $$MODSYMS.tmp.new $$MODSYMS.diff Nothing_to_remove


355 $(OBJS_DIR):
356         -@mkdir -p $@ 2> /dev/null

358 def.targ:               $(OBJS_DIR) $(ALL_TARGET)

360 all.targ:               $(OBJS_DIR) $(ALL_TARGET)

362 lint.targ:              $(OBJS_DIR) $(LINT_TARGET)

364 modlintlib.targ:        $(OBJS_DIR) $(MOD_LINT_LIB)

366 install.targ:           $(OBJS_DIR) $(INSTALL_TARGET)

368 #
369 # Support for Install.sh.
370 #

372 modlist:        $(MODLIST_DEPS)

374 # paths relative to $(ROOT).
375 RELMODULE = $(ROOTMODULE:$(ROOT)/%=%)
376 RELCONF = $(ROOT_CONFFILE:$(ROOT)/%=%)
377 RELLINK = $(ROOTLINK:$(ROOT)/%=%)
378 RELUNIX = $(UNIX32_LINK:$(ROOT)/%=%)
379 RELSOFTLINKS = $(ROOTSOFTLINKS:$(ROOT)/%=%)

381 MODSRC:sh=              pwd

383 #
384 # Generate module information for Install.sh, i.e., specify what files
385 # Install.sh should include.  Each line looks like
386 #   <tag> <srcdir> <arg1> <arg2> ...
387 # where <tag> specifies the type of file, <srcdir> gives the source
388 # path (useful if there is an error), and <argN> is one or more
389 # additional bits of information that Install.sh needs (e.g., source
390 # directory, install directory, filtering tags).  See Install.sh for
391 # details on the arguments for each tag type, especially the functions
```

```
392 # copymod, filtmod, and filtimpl.
393 #
394 # Changes to this target may require corresponding changes to
395 # Install.sh.
396 #
397 # Don't issue a MOD entry if it's not in the install list.
398 #

400 $(MODLIST_DEPS): FRC
401         @case $@ in \
402         *32) \
403                 class=32; \
404                 [ -n "$(RELMODULE)" ] && relmodule=`dirname $(RELMODULE)`;; \
405         *64) \
406                 class=64; \
407                 [ -n "$(RELMODULE)" ] && \
408                     relmodule=`dirname $(RELMODULE)`/$(SUBDIR64);; \
409         esac; \
410         if [ -z "$(THISIMPL)" ]; then \
411                 impl=all; \
412         else \
413                 impl=$(THISIMPL); \
414         fi; \
415         if [ -n "$(ROOTMODULE)" -a -n "$(INSTALL_TARGET)" ]; then \
416                 if [ -z "$(MODULE)" ]; then \
417                         module=`basename $(ROOTMODULE)`; \
418                 else \
419                         module=$(MODULE); \
420                 fi; \
421                 tinstall="$(INSTALL_TARGET)"; \
422                 for t in $$tinstall; do \
423                         if [ "$(ROOTMODULE)" = $$t ]; then \
424                                 echo MOD $(MODSRC) $$module $$relmodule \
425                                     $$class $$impl; \
426                                 break; \
427                         fi \
428                 done \
429         fi; \
430         if [ -n "$(CONF_SRCDIR)" ]; then \
431                 tinstall="$(INSTALL_TARGET)"; \
432                 for t in $$tinstall; do \
433                         if [ $(ROOT_CONFFILE) = $$t ]; then \
434                                 echo CONF $(MODSRC) $(RELCONF) \
435                                     $(MODSRC)/$(CONF_SRCDIR) $$impl $$module; \
436                                 break; \
437                         fi \
438                 done \
439         fi; \
440         if [ -n "$(ROOTLINK)" ]; then \
441                 rellinks="$(RELLINK)"; \
442                 for r in $$rellinks; do \
443                         if [ $$class = 32 ]; then \
444                                 linkdir=`dirname $$r`; \
445                         else \
446                                 linkdir=`dirname $$r`/$(SUBDIR64); \
447                         fi; \
448                         echo LINK $(MODSRC) $$relmodule $$module \
449                                 $$linkdir `basename $$r` $$impl; \
450                 done \
451         fi; \
452         if [ -n "$(UNIX32_LINK)" ]; then \
453                 echo SYMLINK $(MODSRC) $(SUBDIR64)/$(UNIX) \
454                     `dirname $(RELUNIX)` unix $$impl $$module; \
455         fi; \
456         trelsoftlinks="$(RELSOFTLINKS)"; \
457         for t in $$trelsoftlinks; do \
```



```
458                 if [ $$class = 32 ]; then \
459                         linkdir=`dirname $$t`; \
460                 else \
461                         linkdir=`dirname $$t`/$(SUBDIR64); \
462                 fi; \
463                 linkname=`basename $$t`; \
464                 echo SYMLINK $(MODSRC) $(MODULE) $$linkdir $$linkname \
465                     $$impl $$module; \
466         done

468 #
469 #       Cleanliness is next to ...
470 #
471 clean.targ:
472         -$(RM) $(CLEANFILES) Nothing_to_remove

474 clobber.targ:
475         -$(RM) $(CLOBBERFILES) Nothing_to_remove

477 clean.lint.targ:
478         -$(RM) $(CLEANLINTFILES) Nothing_to_remove

480 #
481 #       Create fake lintlibs in the 64b dirs so
482 #       global linting works
483 #
484 lint64:
485         @$(ECHO) $(MODULE) fake lints
486         @for dir in $(LINT64_DIRS); do \
487                 if [ ! -d $$dir ]; then mkdir $$dir; fi \
488         done
489         @for file in $(LINT64_FILES); do \
490                 if [ ! -f $$file ]; then touch $$file; fi \
491         done

493 #
494 #       In some places we also need to create fake lintlibs for 32b
495 #       dirs so global linting works
496 #
497 lint32:
498         @$(ECHO) $(MODULE) fake lints
499         @for dir in $(LINT32_DIRS); do \
500                 if [ ! -d $$dir ]; then mkdir $$dir; fi \
501         done
502         @for file in $(LINT32_FILES); do \
503                 if [ ! -f $$file ]; then touch $$file; fi \
504         done

506 FRC:
```