```
**********************************************************
   35221 Sat Jun 18 19:37:52 2016
new/usr/src/Makefile.master
7069 i386_XARCH should include -m32
**********************************************************
```

```
#
# 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 2015, OmniTI Computer Consulting, Inc. All rights reserved.
# Copyright 2015 Gary Mills
# Copyright 2015 Igor Kozhukhov <ikozhukhov@gmail.com>
#

#
# 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)

# Allow build-time "configuration" to enable or disable some things.
# The default is POUND_SIGN, meaning "not enabled". If the environment
# passes in an override like ENABLE_SMB_PRINTING= (empty) that will
# uncomment things in the lower Makefiles to enable the feature.
ENABLE_IPP_PRINTING=    $(POUND_SIGN)
ENABLE_SMB_PRINTING=    $(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

# define runtime JAVA_HOME, primarily for cmd/pools/poold
JAVA_HOME=      /usr/java
# define buildtime JAVA_ROOT
JAVA_ROOT=      /usr/java

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
```

```
129 RPCGEN=         /usr/bin/rpcgen
130 STABS=          $(ONBLD_TOOLS)/bin/$(MACH)/stabs
131 ELFEXTRACT=     $(ONBLD_TOOLS)/bin/$(MACH)/elfextract
132 MBH_PATCH=      $(ONBLD_TOOLS)/bin/$(MACH)/mbh_patch
133 ECHO=           echo
134 INS=            install
135 TRUE=           true
136 SYMLINK=        /usr/bin/ln -s
137 LN=             /usr/bin/ln
138 CHMOD=          /usr/bin/chmod
139 MV=             /usr/bin/mv -f
140 RM=             /usr/bin/rm -f
141 CUT=            /usr/bin/cut
142 NM=             /usr/ccs/bin/nm
143 DIFF=           /usr/bin/diff
144 GREP=           /usr/bin/grep
145 EGREP=          /usr/bin/egrep
146 ELFWRAP=        /usr/bin/elfwrap
147 KSH93=          /usr/bin/ksh93
148 SED=            /usr/bin/sed
149 AWK=            /usr/bin/nawk
150 CP=             /usr/bin/cp -f
151 MCS=            /usr/ccs/bin/mcs
152 CAT=            /usr/bin/cat
153 ELFDUMP=        /usr/ccs/bin/elfdump
154 M4=             /usr/bin/m4
155 STRIP=          /usr/ccs/bin/strip
156 LEX=            /usr/ccs/bin/lex
157 FLEX=           /usr/bin/flex
158 YACC=           /usr/ccs/bin/yacc
159 CPP=            /usr/lib/cpp
160 JAVAC=          $(JAVA_ROOT)/bin/javac
161 JAVAH=          $(JAVA_ROOT)/bin/javah
162 JAVADOC=        $(JAVA_ROOT)/bin/javadoc
163 RMIC=           $(JAVA_ROOT)/bin/rmic
164 JAR=            $(JAVA_ROOT)/bin/jar
165 CTFCONVERT=     $(ONBLD_TOOLS)/bin/$(MACH)/ctfconvert
166 CTFMERGE=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfmerge
167 CTFSTABS=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfstabs
168 CTFSTRIP=       $(ONBLD_TOOLS)/bin/$(MACH)/ctfstrip
169 NDRGEN=         $(ONBLD_TOOLS)/bin/$(MACH)/ndrgen
170 GENOFFSETS=     $(ONBLD_TOOLS)/bin/genoffsets
171 XREF=           $(ONBLD_TOOLS)/bin/xref
172 FIND=           /usr/bin/find
173 PERL=           /usr/bin/perl
174 PERL_VERSION=   5.10.0
175 PERL_PKGVERS=   -510
176 PERL_ARCH =             i86pc-solaris-64int
177 $(SPARC_BLD)PERL_ARCH = sun4-solaris-64int
178 PYTHON_26=      /usr/bin/python2.6
179 PYTHON=         $(PYTHON_26)
180 SORT=           /usr/bin/sort
181 TOUCH=          /usr/bin/touch
182 WC=             /usr/bin/wc
183 XARGS=          /usr/bin/xargs
184 ELFEDIT=        /usr/bin/elfedit
185 ELFSIGN=        /usr/bin/elfsign
186 DTRACE=         /usr/sbin/dtrace -xnolibs
187 UNIQ=           /usr/bin/uniq
188 TAR=            /usr/bin/tar
189 ASTBINDIR=      /usr/ast/bin
190 MSGCC=          $(ASTBINDIR)/msgcc
191 MSGFMT=         /usr/bin/msgfmt -s

193 FILEMODE=       644
```



```
194 DIRMODE=        755

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

200 # For stylistic checks
201 #
202 # Note that the X and C checks are not used at this time and may need
203 # modification when they are actually used.
204 #
205 CSTYLE=         $(ONBLD_TOOLS)/bin/cstyle
206 CSTYLE_TAIL=
207 HDRCHK=         $(ONBLD_TOOLS)/bin/hdrchk
208 HDRCHK_TAIL=
209 JSTYLE=         $(ONBLD_TOOLS)/bin/jstyle

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

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

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

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

229 INS.file=       $(RM) $@; $(INS) -s -m $(FILEMODE) -f $(@D) $<
230 INS.dir=        $(INS) -s -d -m $(DIRMODE) $@
231 # installs and renames at once
232 #
233 INS.rename=     $(INS.file); $(MV) $(@D)/$(<F) $@

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

240 #
241 # Python bakes the mtime of the .py file into the compiled .pyc and
242 # rebuilds if the baked-in mtime != the mtime of the source file
243 # (rather than only if it's less than), thus when installing python
244 # files we must make certain to not adjust the mtime of the source
245 # (.py) file.
246 #
247 INS.pyfile=     $(INS.file); $(TOUCH) -r $< $@

249 # MACH must be set in the shell environment per uname -p on the build host
250 # More specific architecture variables should be set in lower makefiles.
251 #
252 # MACH64 is derived from MACH, and BUILD64 is set to '#' for
253 # architectures on which we do not build 64-bit versions.
254 # (There are no such architectures at the moment.)
255 #
256 # Set BUILD64=# in the environment to disable 64-bit amd64
257 # builds on i386 machines.

259 MACH64_1=       $(MACH:sparc=sparcv9)
```

```
260 MACH64=         $(MACH64_1:i386=amd64)

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

265 sparc_BUILD64=
266 i386_BUILD64=
267 BUILD64=        $($(MACH)_BUILD64)

269 #
270 # C compiler mode. Future compilers may change the default on us,
271 # so force extended ANSI mode globally. Lower level makefiles can
272 # override this by setting CCMODE.
273 #
274 CCMODE=                 -Xa
275 CCMODE64=               -Xa

277 #
278 # C compiler verbose mode. This is so we can enable it globally,
279 # but turn it off in the lower level makefiles of things we cannot
280 # (or aren't going to) fix.
281 #
282 CCVERBOSE=              -v

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

288 # set this to the secret flag "-Wc,-Qiselect-regsym=0" to disable register
289 # symbols (used to detect conflicts between objects that use global registers)
290 # we disable this now for safety, and because genunix doesn't link with
291 # this feature (the v9 default) enabled.
292 #
293 # REGSYM is separate since the C++ driver syntax is different.
294 CCREGSYM=               -Wc,-Qiselect-regsym=0
295 CCCREGSYM=              -Qoption cg -Qiselect-regsym=0

297 # Prevent the removal of static symbols by the SPARC code generator (cg).
298 # The x86 code generator (ube) does not remove such symbols and as such
299 # using this workaround is not applicable for x86.
300 #
301 CCSTATICSYM=            -Wc,-Qassembler-ounrefsym=0
302 #
303 # generate 32-bit addresses in the v9 kernel. Saves memory.
304 CCABS32=                -Wc,-xcode=abs32
305 #
306 # generate v9 code which tolerates callers using the v7 ABI, for the sake of
307 # system calls.
308 CC32BITCALLERS=         -_gcc=-massume-32bit-callers

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

319 # One optimization the compiler might perform is to turn this:
320 #       #pragma weak foo
321 #       extern int foo;
322 #       if (&foo)
323 #               foo = 5;
324 # into
325 #       foo = 5;
```



```
326 # Since we do some of this (foo might be referenced in common kernel code
327 # but provided only for some cpu modules or platforms), we disable this
328 # optimization.
329 #
330 sparc_CCUNBOUND = -Wd,-xsafe=unboundsym
331 i386_CCUNBOUND  =
332 CCUNBOUND       = $($(MACH)_CCUNBOUND)

334 #
335 # compiler '-xarch' flag. This is here to centralize it and make it
336 # overridable for testing.
337 sparc_XARCH=    -m32
338 sparcv9_XARCH=  -m64
339 i386_XARCH=     -m32
339 i386_XARCH=
340 amd64_XARCH=    -m64 -Ui386 -U__i386

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

348 #
349 # These flags define what we need to be 'standalone' i.e. -not- part
350 # of the rather more cosy userland environment.  This basically means
351 # the kernel.
352 #
353 # XX64  future versions of gcc will make -mcmodel=kernel imply -mno-red-zone
354 #
355 sparc_STAND_FLAGS=      -_gcc=-ffreestanding
356 sparcv9_STAND_FLAGS=    -_gcc=-ffreestanding
357 # Disabling MMX also disables 3DNow, disabling SSE also disables all later
358 # additions to SSE (SSE2, AVX ,etc.)
359 NO_SIMD=                -_gcc=-mno-mmx -_gcc=-mno-sse
360 i386_STAND_FLAGS=       -_gcc=-ffreestanding $(NO_SIMD)
361 amd64_STAND_FLAGS=      -xmodel=kernel $(NO_SIMD)

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

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

369 #
370 # disable the incremental linker
371 ILDOFF=                 -xildoff
372 #
373 XDEPEND=                -xdepend
374 XFFLAG=                 -xF=%all
375 XESS=                   -xs
376 XSTRCONST=              -xstrconst

378 #
379 # turn warnings into errors (C)
380 CERRWARN = -erttags=yes -errwarn=%all
381 CERRWARN += -erroff=E_EMPTY_TRANSLATION_UNIT
382 CERRWARN += -erroff=E_STATEMENT_NOT_REACHED

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

390 # Unfortunately, this option can misfire very easily and unfixably.
```

```
391 CERRWARN +=     -_gcc=-Wno-array-bounds

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

398 #
399 # turn warnings into errors (C++)
400 CCERRWARN=              -xwe

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

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

416 sparc_ASFLAGS=  $(sparc_AS_XARCH)
417 sparcv9_ASFLAGS=$(sparcv9_AS_XARCH)
418 i386_ASFLAGS=   $(i386_AS_XARCH)
419 amd64_ASFLAGS=  $(amd64_AS_XARCH)

421 #
422 sparc_COPTFLAG=         -xO3
423 sparcv9_COPTFLAG=       -xO3
424 i386_COPTFLAG=          -O
425 amd64_COPTFLAG=         -xO3

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

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

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

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

446 #
447 # Default debug format for Sun Studio 11 is dwarf, so force it to
448 # generate stabs.
449 #
450 DEBUGFORMAT=    -xdebugformat=stabs

452 #
453 # Flags used to build in debug mode for ctf generation.  Bugs in the Devpro
454 # compilers currently prevent us from building with cc-emitted DWARF.
455 #
456 CTF_FLAGS_sparc = -g -Wc,-Qiselect-T1 $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)
```



```
457 CTF_FLAGS_i386  = -g $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)

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

462 # Sun Studio produces broken userland code when saving arguments.
463 $(__GNUC)CTF_FLAGS_amd64 += $(SAVEARGS)

465 CTF_FLAGS_32    = $(CTF_FLAGS_$(MACH)) $(DEBUGFORMAT)
466 CTF_FLAGS_64    = $(CTF_FLAGS_$(MACH64)) $(DEBUGFORMAT)
467 CTF_FLAGS       = $(CTF_FLAGS_32)

469 #
470 # Flags used with genoffsets
471 #
472 GOFLAGS = -_noecho \
473         $(CALLSYMS) \
474         $(CDWARFSTR)

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

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

482 #
483 # tradeoff time for space (smaller is better)
484 #
485 sparc_SPACEFLAG         = -xspace -W0,-Lt
486 sparcv9_SPACEFLAG       = -xspace -W0,-Lt
487 i386_SPACEFLAG          = -xspace
488 amd64_SPACEFLAG         =

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

493 #
494 # The Sun Studio 11 compiler has changed the behaviour of integer
495 # wrap arounds and so a flag is needed to use the legacy behaviour
496 # (without this flag panics/hangs could be exposed within the source).
497 #
498 sparc_IROPTFLAG         = -W2,-xwrap_int
499 sparcv9_IROPTFLAG       = -W2,-xwrap_int
500 i386_IROPTFLAG          =
501 amd64_IROPTFLAG         =

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

506 sparc_XREGSFLAG         = -xregs=no%appl
507 sparcv9_XREGSFLAG       = -xregs=no%appl
508 i386_XREGSFLAG          =
509 amd64_XREGSFLAG         =

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

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

519 #
520 # These variables are intended ONLY for use by developers to safely pass extra
521 # flags to the compilers without unintentionally overriding Makefile-set
522 # flags.  They should NEVER be set to any value in a Makefile.
```

```
523 #
524 # They come last in the associated FLAGS variable such that they can
525 # explicitly override things if necessary, there are gaps in this, but it's
526 # the best we can manage.
527 #
528 CUSERFLAGS              =
529 CUSERFLAGS64            = $(CUSERFLAGS)
530 CCUSERFLAGS             =
531 CCUSERFLAGS64           = $(CCUSERFLAGS)

533 CSOURCEDEBUGFLAGS       =
534 CCSOURCEDEBUGFLAGS      =
535 $(SRCDBGBLD)CSOURCEDEBUGFLAGS   = -g -xs
536 $(SRCDBGBLD)CCSOURCEDEBUGFLAGS  = -g -xs

538 CFLAGS=         $(COPTFLAG) $($(MACH)_CFLAGS) $(SPACEFLAG) $(CCMODE) \
539                 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG) \
540                 $(CGLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
541                 $(CUSERFLAGS)
542 CFLAGS64=       $(COPTFLAG64) $($(MACH64)_CFLAGS) $(SPACEFLAG64) $(CCMODE64) \
543                 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG64) \
544                 $(CGLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
545                 $(CUSERFLAGS64)
546 #
547 # Flags that are used to build parts of the code that are subsequently
548 # run on the build machine (also known as the NATIVE_BUILD).
549 #
550 NATIVE_CFLAGS=  $(COPTFLAG) $($(NATIVE_MACH)_CFLAGS) $(CCMODE) \
551                 $(ILDOFF) $(CERRWARN) $(C99MODE) $($(NATIVE_MACH)_CCUNBOUND) \
552                 $(IROPTFLAG) $(CGLOBALSTATIC) $(CCNOAUTOINLINE) \
553                 $(CSOURCEDEBUGFLAGS) $(CUSERFLAGS)

555 DTEXTDOM=-DTEXT_DOMAIN=\"$(TEXT_DOMAIN)\"       # For messaging.
556 DTS_ERRNO=-D_TS_ERRNO
557 CPPFLAGS.first= # Please keep empty.  Only lower makefiles should set this.
558 CPPFLAGS.master=$(DTEXTDOM) $(DTS_ERRNO) \
559         $(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) $(ENVCPPFLAGS4) \
560         $(ADJUNCT_PROTO:%=-I%/usr/include)
561 CPPFLAGS.native=$(ENVCPPFLAGS1) $(ENVCPPFLAGS2) $(ENVCPPFLAGS3) \
562                 $(ENVCPPFLAGS4) -I$(NATIVE_ADJUNCT)/include
563 CPPFLAGS=       $(CPPFLAGS.first) $(CPPFLAGS.master)
564 AS_CPPFLAGS=    $(CPPFLAGS.first) $(CPPFLAGS.master)
565 JAVAFLAGS=      -source 1.6 -target 1.6 -Xlint:deprecation,-options

567 #
568 # For source message catalogue
569 #
570 .SUFFIXES: $(SUFFIXES) .i .po
571 MSGROOT= $(ROOT)/catalog
572 MSGDOMAIN= $(MSGROOT)/$(TEXT_DOMAIN)
573 MSGDOMAINPOFILE = $(MSGDOMAIN)/$(POFILE)
574 DCMSGDOMAIN= $(MSGROOT)/LC_TIME/$(TEXT_DOMAIN)
575 DCMSGDOMAINPOFILE = $(DCMSGDOMAIN)/$(DCFILE:.dc=.po)

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



```
589 #
590 # This is overwritten by local Makefile when PROG is a list.
591 #
592 POFILE= $(PROG).po

594 sparc_CCFLAGS=          -cg92 -compat=4 \
595                         -Qoption ccfe -messages=no%anachronism \
596                         $(CCERRWARN)
597 sparcv9_CCFLAGS=        $(sparcv9_XARCH) -dalign -compat=5 \
598                         -Qoption ccfe -messages=no%anachronism \
599                         -Qoption ccfe -features=no%conststrings \
600                         $(CCCREGSYM) \
601                         $(CCERRWARN)
602 i386_CCFLAGS=           -compat=4 \
603                         -Qoption ccfe -messages=no%anachronism \
604                         -Qoption ccfe -features=no%conststrings \
605                         $(CCERRWARN)
606 amd64_CCFLAGS=          $(amd64_XARCH) -compat=5 \
607                         -Qoption ccfe -messages=no%anachronism \
608                         -Qoption ccfe -features=no%conststrings \
609                         $(CCERRWARN)

611 sparc_CCOPTFLAG=        -O
612 sparcv9_CCOPTFLAG=      -O
613 i386_CCOPTFLAG=         -O
614 amd64_CCOPTFLAG=        -O

616 CCOPTFLAG=      $($(MACH)_CCOPTFLAG)
617 CCOPTFLAG64=    $($(MACH64)_CCOPTFLAG)
618 CCFLAGS=        $(CCOPTFLAG) $($(MACH)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
619                 $(CCUSERFLAGS)
620 CCFLAGS64=      $(CCOPTFLAG64) $($(MACH64)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
621                 $(CCUSERFLAGS64)

623 #
624 #
625 #
626 ELFWRAP_FLAGS   =
627 ELFWRAP_FLAGS64 =       -64

629 #
630 # Various mapfiles that are used throughout the build, and delivered to
631 # /usr/lib/ld.
632 #
633 MAPFILE.NED_i386 =      $(SRC)/common/mapfiles/common/map.noexdata
634 MAPFILE.NED_sparc =
635 MAPFILE.NED =           $(MAPFILE.NED_$(MACH))
636 MAPFILE.PGA =           $(SRC)/common/mapfiles/common/map.pagealign
637 MAPFILE.NES =           $(SRC)/common/mapfiles/common/map.noexstk
638 MAPFILE.FLT =           $(SRC)/common/mapfiles/common/map.filter
639 MAPFILE.LEX =           $(SRC)/common/mapfiles/common/map.lex.yy

641 #
642 # Generated mapfiles that are compiler specific, and used throughout the
643 # build.  These mapfiles are not delivered in /usr/lib/ld.
644 #
645 MAPFILE.NGB_sparc=      $(SRC)/common/mapfiles/gen/sparc_cc_map.noexeglobs
646 $(__GNUC64)MAPFILE.NGB_sparc= \
647                         $(SRC)/common/mapfiles/gen/sparc_gcc_map.noexeglobs
648 MAPFILE.NGB_sparcv9=    $(SRC)/common/mapfiles/gen/sparcv9_cc_map.noexeglobs
649 $(__GNUC64)MAPFILE.NGB_sparcv9= \
650                         $(SRC)/common/mapfiles/gen/sparcv9_gcc_map.noexeglobs
651 MAPFILE.NGB_i386=       $(SRC)/common/mapfiles/gen/i386_cc_map.noexeglobs
652 $(__GNUC64)MAPFILE.NGB_i386= \
653                         $(SRC)/common/mapfiles/gen/i386_gcc_map.noexeglobs
654 MAPFILE.NGB_amd64=      $(SRC)/common/mapfiles/gen/amd64_cc_map.noexeglobs
```

```
655 $(__GNUC64)MAPFILE.NGB_amd64= \
656                         $(SRC)/common/mapfiles/gen/amd64_gcc_map.noexeglobs
657 MAPFILE.NGB =           $(MAPFILE.NGB_$(MACH))

659 #
660 # A generic interface mapfile name, used by various dynamic objects to define
661 # the interfaces and interposers the object must export.
662 #
663 MAPFILE.INT =           mapfile-intf

665 #
666 # LDLIBS32 and LDLIBS64 can be set in the environment to override the following
667 # assignments.
668 #
669 # These environment settings make sure that no libraries are searched outside
670 # of the local workspace proto area:
671 #       LDLIBS32=-YP,$ROOT/lib:$ROOT/usr/lib
672 #       LDLIBS64=-YP,$ROOT/lib/$MACH64:$ROOT/usr/lib/$MACH64
673 #
674 LDLIBS32 =      $(ENVLDLIBS1) $(ENVLDLIBS2) $(ENVLDLIBS3)
675 LDLIBS32 +=     $(ADJUNCT_PROTO:%=-L%/usr/lib -L%/lib)
676 LDLIBS.cmd =    $(LDLIBS32)
677 LDLIBS.lib =    $(LDLIBS32)

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

684 #
685 # Define compilation macros.
686 #
687 COMPILE.c=      $(CC) $(CFLAGS) $(CPPFLAGS) -c
688 COMPILE64.c=    $(CC) $(CFLAGS64) $(CPPFLAGS) -c
689 COMPILE.cc=     $(CCC) $(CCFLAGS) $(CPPFLAGS) -c
690 COMPILE64.cc=   $(CCC) $(CCFLAGS64) $(CPPFLAGS) -c
691 COMPILE.s=      $(AS) $(ASFLAGS) $(AS_CPPFLAGS)
692 COMPILE64.s=    $(AS) $(ASFLAGS) $($(MACH64)_AS_XARCH) $(AS_CPPFLAGS)
693 COMPILE.d=      $(DTRACE) -G -32
694 COMPILE64.d=    $(DTRACE) -G -64
695 COMPILE.b=      $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))
696 COMPILE64.b=    $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))

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

701 #
702 # Link time macros
703 #
704 CCNEEDED                = -lC
705 CCEXTNEEDED             = -lCrun -lCstd
706 $(__GNUC)CCNEEDED       = -L$(GCCLIBDIR) -lstdc++ -lgcc_s
707 $(__GNUC)CCEXTNEEDED    = $(CCNEEDED)

709 LINK.c=         $(CC) $(CFLAGS) $(CPPFLAGS) $(LDFLAGS)
710 LINK64.c=       $(CC) $(CFLAGS64) $(CPPFLAGS) $(LDFLAGS)
711 NORUNPATH=      -norunpath -nolib
712 LINK.cc=        $(CCC) $(CCFLAGS) $(CPPFLAGS) $(NORUNPATH) \
713                 $(LDFLAGS) $(CCNEEDED)
714 LINK64.cc=      $(CCC) $(CCFLAGS64) $(CPPFLAGS) $(NORUNPATH) \
715                 $(LDFLAGS) $(CCNEEDED)

717 #
718 # lint macros
719 #
720 # Note that the undefine of __PRAGMA_REDEFINE_EXTNAME can be removed once
```



```
721 # ON is built with a version of lint that has the fix for 4484186.
722 #
723 ALWAYS_LINT_DEFS =      -errtags=yes -s
724 ALWAYS_LINT_DEFS +=     -erroff=E_PTRDIFF_OVERFLOW
725 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_NARROW_CONV
726 ALWAYS_LINT_DEFS +=     -U__PRAGMA_REDEFINE_EXTNAME
727 ALWAYS_LINT_DEFS +=     $(C99LMODE)
728 ALWAYS_LINT_DEFS +=     -errsecurity=$(SECLEVEL)
729 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_CREAT_WITHOUT_EXCL
730 ALWAYS_LINT_DEFS +=     -erroff=E_SEC_FORBIDDEN_WARN_CREAT
731 # XX64 -- really only needed for amd64 lint
732 ALWAYS_LINT_DEFS +=     -erroff=E_ASSIGN_INT_TO_SMALL_INT
733 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_CONST_TO_SMALL_INT
734 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_INT_TO_SMALL_INT
735 ALWAYS_LINT_DEFS +=     -erroff=E_CAST_TO_PTR_FROM_INT
736 ALWAYS_LINT_DEFS +=     -erroff=E_COMP_INT_WITH_LARGE_INT
737 ALWAYS_LINT_DEFS +=     -erroff=E_INTEGRAL_CONST_EXP_EXPECTED
738 ALWAYS_LINT_DEFS +=     -erroff=E_PASS_INT_TO_SMALL_INT
739 ALWAYS_LINT_DEFS +=     -erroff=E_PTR_CONV_LOSES_BITS

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

745 SECLEVEL=       core
746 LINT.c=         $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS) $(CPPFLAGS) \
747                 $(ALWAYS_LINT_DEFS)
748 LINT64.c=       $(LINT) $(ONLY_LINT_DEFS) $(LINTFLAGS64) $(CPPFLAGS) \
749                 $(ALWAYS_LINT_DEFS)
750 LINT.s=         $(LINT.c)

752 # For some future builds, NATIVE_MACH and MACH might be different.
753 # Therefore, NATIVE_MACH needs to be redefined in the
754 # environment as ‘uname -p‘ to override this macro.
755 #
756 # For now at least, we cross-compile amd64 on i386 machines.
757 NATIVE_MACH=    $(MACH:amd64=i386)

759 # Define native compilation macros
760 #

762 # Base directory where compilers are loaded.
763 # Defined here so it can be overridden by developer.
764 #
765 SPRO_ROOT=              $(BUILD_TOOLS)/SUNWspro
766 SPRO_VROOT=             $(SPRO_ROOT)/SS12
767 GNU_ROOT=               /usr

769 # Till SS12u1 formally becomes the NV CBE, LINT is hard
770 # coded to be picked up from the $SPRO_ROOT/sunstudio12.1/
771 # location. Impacted variables are sparc_LINT, sparcv9_LINT,
772 # i386_LINT, amd64_LINT.
773 # Reset them when SS12u1 is rolled out.
774 #

776 # Specify platform compiler versions for languages
777 # that we use (currently only c and c++).
778 #
779 sparc_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
780 $(__GNUC)sparc_CC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
781 sparc_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
782 $(__GNUC)sparc_CCC=     $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
783 sparc_CPP=              /usr/ccs/lib/cpp
784 sparc_AS=               /usr/ccs/bin/as -xregsym=no
785 sparc_LD=               /usr/ccs/bin/ld
786 sparc_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint
```

```
788 sparcv9_CC=             $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
789 $(__GNUC64)sparcv9_CC=  $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
790 sparcv9_CCC=            $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
791 $(__GNUC64)sparcv9_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
792 sparcv9_CPP=            /usr/ccs/lib/cpp
793 sparcv9_AS=             /usr/ccs/bin/as -xregsym=no
794 sparcv9_LD=             /usr/ccs/bin/ld
795 sparcv9_LINT=           $(SPRO_ROOT)/sunstudio12.1/bin/lint

797 i386_CC=               $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
798 $(__GNUC)i386_CC=       $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
799 i386_CCC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
800 $(__GNUC)i386_CCC=      $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
801 i386_CPP=              /usr/ccs/lib/cpp
802 i386_AS=               /usr/ccs/bin/as
803 $(__GNUC)i386_AS=       $(ONBLD_TOOLS)/bin/$(MACH)/aw
804 i386_LD=               /usr/ccs/bin/ld
805 i386_LINT=             $(SPRO_ROOT)/sunstudio12.1/bin/lint

807 amd64_CC=              $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
808 $(__GNUC64)amd64_CC=    $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
809 amd64_CCC=             $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
810 $(__GNUC64)amd64_CCC=   $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
811 amd64_CPP=             /usr/ccs/lib/cpp
812 amd64_AS=              $(ONBLD_TOOLS)/bin/$(MACH)/aw
813 amd64_LD=              /usr/ccs/bin/ld
814 amd64_LINT=            $(SPRO_ROOT)/sunstudio12.1/bin/lint

816 NATIVECC=              $($(NATIVE_MACH)_CC)
817 NATIVECCC=             $($(NATIVE_MACH)_CCC)
818 NATIVECPP=             $($(NATIVE_MACH)_CPP)
819 NATIVEAS=              $($(NATIVE_MACH)_AS)
820 NATIVELD=              $($(NATIVE_MACH)_LD)
821 NATIVELINT=            $($(NATIVE_MACH)_LINT)

823 #
824 # Makefile.master.64 overrides these settings
825 #
826 CC=                    $(NATIVECC)
827 CCC=                   $(NATIVECCC)
828 CPP=                   $(NATIVECPP)
829 AS=                    $(NATIVEAS)
830 LD=                    $(NATIVELD)
831 LINT=                  $(NATIVELINT)

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

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

842 BDIRECT=        -Bdirect
843 BDYNAMIC=       -Bdynamic
844 BLOCAL=         -Blocal
845 BNODIRECT=      -Bnodirect
846 BREDUCE=        -Breduce
847 BSTATIC=        -Bstatic

849 ZDEFS=          -zdefs
850 ZDIRECT=        -zdirect
851 ZIGNORE=        -zignore
852 ZINITFIRST=     -zinitfirst
```



```
853 ZINTERPOSE=     -zinterpose
854 ZLAZYLOAD=      -zlazyload
855 ZLOADFLTR=      -zloadfltr
856 ZMULDEFS=       -zmuldefs
857 ZNODEFAULTLIB=  -znodefaultlib
858 ZNODEFS=        -znodefs
859 ZNODELETE=      -znodelete
860 ZNODLOPEN=      -znodlopen
861 ZNODUMP=        -znodump
862 ZNOLAZYLOAD=    -znolazyload
863 ZNOLDYNSYM=     -znoldynsym
864 ZNORELOC=       -znoreloc
865 ZNOVERSION=     -znoversion
866 ZRECORD=        -zrecord
867 ZREDLOCSYM=     -zredlocsym
868 ZTEXT=          -ztext
869 ZVERBOSE=       -zverbose

871 GSHARED=        -G
872 CCMT=           -mt

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

877 sparc_C_PICFLAGS =      -K pic
878 sparcv9_C_PICFLAGS =    -K pic
879 i386_C_PICFLAGS =       -K pic
880 amd64_C_PICFLAGS =      -K pic
881 C_PICFLAGS =            $($(MACH)_C_PICFLAGS)
882 C_PICFLAGS64 =          $($(MACH64)_C_PICFLAGS)

884 sparc_C_BIGPICFLAGS =   -K PIC
885 sparcv9_C_BIGPICFLAGS = -K PIC
886 i386_C_BIGPICFLAGS =    -K PIC
887 amd64_C_BIGPICFLAGS =   -K PIC
888 C_BIGPICFLAGS =         $($(MACH)_C_BIGPICFLAGS)
889 C_BIGPICFLAGS64 =       $($(MACH64)_C_BIGPICFLAGS)

891 # CC requires there to be no space between '-K' and 'pic' or 'PIC'.
892 sparc_CC_PICFLAGS =     -Kpic
893 sparcv9_CC_PICFLAGS =   -KPIC
894 i386_CC_PICFLAGS =      -Kpic
895 amd64_CC_PICFLAGS =     -Kpic
896 CC_PICFLAGS =           $($(MACH)_CC_PICFLAGS)
897 CC_PICFLAGS64 =         $($(MACH64)_CC_PICFLAGS)

899 AS_PICFLAGS=            $(C_PICFLAGS)
900 AS_BIGPICFLAGS=         $(C_BIGPICFLAGS)

902 #
903 # Default label for CTF sections
904 #
905 CTFCVTFLAGS=            -i -L VERSION

907 #
908 # Override to pass module-specific flags to ctfmerge.  Currently used only by
909 # krtld to turn on fuzzy matching, and source-level debugging to inhibit
910 # stripping.
911 #
912 CTFMRGFLAGS=

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

916 ELFSIGN_O=      $(TRUE)
917 ELFSIGN_CRYPTO= $(ELFSIGN_O)
918 ELFSIGN_OBJECT= $(ELFSIGN_O)
```

```
920 # Rules (normally from make.rules) and macros which are used for post
921 # processing files. Normally, these do stripping of the comment section
922 # automatically.
923 #    RELEASE_CM:        Should be editted to reflect the release.
924 #    POST_PROCESS_O:    Post-processing for '.o' files.
925 #    POST_PROCESS_A:    Post-processing for '.a' files (currently null).
926 #    POST_PROCESS_SO:   Post-processing for '.so' files.
927 #    POST_PROCESS:      Post-processing for executable files (no suffix).
928 # Note that these macros are not completely generalized as they are to be
929 # used with the file name to be processed following.
930 #
931 # It is left as an exercise to Release Engineering to embellish the generation
932 # of the release comment string.
933 #
934 #       If this is a standard development build:
935 #               compress the comment section (mcs -c)
936 #               add the standard comment (mcs -a $(RELEASE_CM))
937 #               add the development specific comment (mcs -a $(DEV_CM))
938 #
939 #       If this is an installation build:
940 #               delete the comment section (mcs -d)
941 #               add the standard comment (mcs -a $(RELEASE_CM))
942 #               add the development specific comment (mcs -a $(DEV_CM))
943 #
944 #       If this is an release build:
945 #               delete the comment section (mcs -d)
946 #               add the standard comment (mcs -a $(RELEASE_CM))
947 #
948 # The following list of macros are used in the definition of RELEASE_CM
949 # which is used to label all binaries in the build:
950 #
951 #       RELEASE         Specific release of the build, eg: 5.2
952 #       RELEASE_MAJOR   Major version number part of $(RELEASE)
953 #       RELEASE_MINOR   Minor version number part of $(RELEASE)
954 #       VERSION         Version of the build (alpha, beta, Generic)
955 #       PATCHID         If this is a patch this value should contain
956 #                       the patchid value (eg: "Generic 100832-01"), otherwise
957 #                       it will be set to $(VERSION)
958 #       RELEASE_DATE    Date of the Release Build
959 #       PATCH_DATE      Date the patch was created, if this is blank it
960 #                       will default to the RELEASE_DATE
961 #
962 RELEASE_MAJOR=  5
963 RELEASE_MINOR=  11
964 RELEASE=        $(RELEASE_MAJOR).$(RELEASE_MINOR)
965 VERSION=        SunOS Development
966 PATCHID=        $(VERSION)
967 RELEASE_DATE=   release date not set
968 PATCH_DATE=     $(RELEASE_DATE)
969 RELEASE_CM=     "@($(POUND_SIGN))SunOS $(RELEASE) $(PATCHID) $(PATCH_DATE)"
970 DEV_CM=         "@($(POUND_SIGN))SunOS Internal Development: non-nightly build"

972 PROCESS_COMMENT=  @?${MCS} -d -a $(RELEASE_CM) -a $(DEV_CM)
973 $(RELEASE_BUILD)PROCESS_COMMENT=   @?${MCS} -d -a $(RELEASE_CM)

975 STRIP_STABS=                       $(STRIP) -x $@
976 $(SRCDBGBLD)STRIP_STABS=           :

978 POST_PROCESS_O=
979 POST_PROCESS_A=
980 POST_PROCESS_SO=        $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
981                         $(ELFSIGN_OBJECT)
982 POST_PROCESS=           $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
983                         $(ELFSIGN_OBJECT)
```



```
 985 #
 986 # chk4ubin is a tool that inspects a module for a symbol table
 987 # ELF section size which can trigger an OBP bug on older platforms.
 988 # This problem affects only specific sun4u bootable modules.
 989 #
 990 CHK4UBIN=       $(ONBLD_TOOLS)/bin/$(MACH)/chk4ubin
 991 CHK4UBINFLAGS=
 992 CHK4UBINARY=    $(CHK4UBIN) $(CHK4UBINFLAGS) $@

 994 #
 995 # PKGARCHIVE specifies the default location where packages should be
 996 # placed if built.
 997 #
 998 $(RELEASE_BUILD)PKGARCHIVESUFFIX=       -nd
 999 PKGARCHIVE=$(SRC)/../../packages/$(MACH)/nightly$(PKGARCHIVESUFFIX)

1001 #
1002 # The repositories will be created with these publisher settings.  To
1003 # update an image to the resulting repositories, this must match the
1004 # publisher name provided to "pkg set-publisher."
1005 #
1006 PKGPUBLISHER_REDIST=    on-nightly
1007 PKGPUBLISHER_NONREDIST= on-extra

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

.cc.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
```

```
**********************************************************
     957 Sat Jun 18 19:37:54 2016
new/usr/src/cmd/cmd-inet/usr.bin/dns-sd/Makefile
7120 mDNS resync was not wsdiff safe
**********************************************************
   1 #
   2 # This file and its contents are supplied under the terms of the
   3 # Common Development and Distribution License ("CDDL"), version 1.0.
   4 # You may only use this file in accordance with the terms of version
   5 # 1.0 of the CDDL.
   6 #
   7 # A full copy of the text of the CDDL should have accompanied this
   8 # source.  A copy of the CDDL is also available via the Internet at
   9 # http://www.illumos.org/license/CDDL.
  10 #

  12 #
  13 # Copyright 2016 Toomas Soome <tsoome@me.com>
  14 #

  16 # cmd/cmd-inet/usr.bin/dns-sd/Makefile

  18 PROG=           dns-sd

  20 include ../../../Makefile.cmd
  21 include ../../Makefile.cmd-inet

  23 OBJS=           ClientCommon.o dns-sd.o
  24 SRCS=           ClientCommon.c dns-sd.c

  26 CFLAGS += $(C99_ENABLE)
  27 CPPFLAGS += -DMDNS_VERSIONSTR_NODTS
  28 #endif /* ! codereview */
  29 LDLIBS += -lsocket -ldns_sd

  31 .KEEP_STATE:

  33 all: $(PROG)

  35 ROOTPROG=   $(PROG:%=$(ROOTBIN)/%)

  37 $(PROG): $(OBJS)
  38         $(LINK.c) $(OBJS) -o $@ $(LDLIBS)
  39         $(POST_PROCESS)

  41 install: all $(ROOTPROG)

  43 clean:
  44         $(RM) $(OBJS)

  46 lint:   lint_SRCS

  48 include ../../../Makefile.targ
```

```
**********************************************************
   75742 Sat Jun 18 19:37:55 2016
new/usr/src/cmd/cmd-inet/usr.bin/dns-sd/dns-sd.c
7120 mDNS resync was not wsdiff safe
**********************************************************
______unchanged_portion_omitted_

1806 // Note: The C preprocessor stringify operator ('#') makes a string from its arg
1807 // e.g. If "version" is #define'd to be "4", then STRINGIFY_AWE(version) will re
1808 // To expand "version" to its value before making the string, use STRINGIFY(vers
1809 #define STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s) # s
1810 #define STRINGIFY(s) STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s)

1812 // NOT static -- otherwise the compiler may optimize it out
1813 // The "@(#) " pattern is a special prefix the "what" command looks for
1814 #ifndef MDNS_VERSIONSTR_NODTS
1815 #endif /* ! codereview */
1816 const char VersionString_SCCS[] = "@(#) dns-sd " STRINGIFY(mDNSResponderVersion)
1817 #else
1818 const char VersionString_SCCS[] = "@(#) dns-sd " STRINGIFY(mDNSResponderVersion)
1819 #endif
1820 #endif /* ! codereview */

1822 #if _BUILDING_XCODE_PROJECT_
1823 // If the process crashes, then this string will be magically included in the au
1824 const char *__crashreporter_info__ = VersionString_SCCS + 5;
1825 asm (".desc ___crashreporter_info__, 0x10");
1826 #endif
```

```
**********************************************************
    1420 Sat Jun 18 19:37:56 2016
new/usr/src/lib/libdns_sd/Makefile.com
7120 mDNS resync was not wsdiff safe
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright 2016 Toomas Soome <tsoome@me.com>
#

LIBRARY =       libdns_sd.a
VERS =          .1
OBJECTS =       dnssd_clientlib.o dnssd_clientstub.o dnssd_ipc.o

include ../../Makefile.lib

LIBS =          $(DYNLIB) $(LINTLIB)
$(LINTLIB):=    SRCS = $(SRCDIR)/$(LINTSRC)

SRCDIR =        ../common

LDLIBS +=       -lsocket -lnsl -lc

C99MODE =       $(C99_ENABLE)
CPPFLAGS +=     -I$(SRCDIR) -DNOT_HAVE_SA_LEN -D_XPG4_2 -D__EXTENSIONS__
CPPFLAGS +=     -DMDNS_VERSIONSTR_NODTS
#endif /* ! codereview */

.PARALLEL =     $(OBJECTS)
.KEEP_STATE:

lint: lintcheck

all: $(LIBS)

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

```
**********************************************************
   12660 Sat Jun 18 19:37:57 2016
new/usr/src/lib/libdns_sd/common/dnssd_clientlib.c
7120 mDNS resync was not wsdiff safe
**********************************************************
______unchanged_portion_omitted_

 350 /*****************************************************************************
 351 *
 352 *   SCCS-compatible version string
 353 *
 354 *****************************************************************************

 356 // For convenience when using the "strings" command, this is the last thing in t

 358 // Note: The C preprocessor stringify operator ('#') makes a string from its arg
 359 // e.g. If "version" is #define'd to be "4", then STRINGIFY_AWE(version) will re
 360 // To expand "version" to its value before making the string, use STRINGIFY(vers
 361 #define STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s) # s
 362 #define STRINGIFY(s) STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s)

 364 // NOT static -- otherwise the compiler may optimize it out
 365 // The "@(#) " pattern is a special prefix the "what" command looks for
 366 #ifndef MDNS_VERSIONSTR_NODTS
 367 #endif /* ! codereview */
 368 const char VersionString_SCCS_libdnssd[] = "@(#) libdns_sd " STRINGIFY(mDNSRespo
 369 #else
 370 const char VersionString_SCCS_libdnssd[] = "@(#) libdns_sd " STRINGIFY(mDNSRespo
 371 #endif
 372 #endif /* ! codereview */
```

```
**********************************************************
    1434 Sat Jun 18 19:37:58 2016
new/usr/src/lib/libdns_sd/java/Makefile.com
7120 mDNS resync was not wsdiff safe
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "%Z%%M% %I%     %E% SMI"

LIBRARY=        libjdns_sd.a
VERS=           .1

OBJECTS=        JNISupport.o

include         $(SRC)/lib/Makefile.lib

LIBS =          $(DYNLIB)

SRCDIR =        ../common

C99MODE =       $(C99_ENABLE)
CPPFLAGS +=     -I$(JAVA_ROOT)/include -I$(JAVA_ROOT)/include/solaris
CPPFLAGS +=     -I../com/apple/dnssd
CPPFLAGS +=     -D_REENTRANT
CPPFLAGS +=     -DMDNS_VERSIONSTR_NODTS
#endif /* ! codereview */

LDLIBS +=       -lc -lsocket -ldns_sd

CLEANFILES=     $(LINTOUT) $(LINTLIB)

LINTLIB =

.KEEP_STATE:

lint:           lintcheck

include         $(SRC)/lib/Makefile.targ

pics/%.o: ../common/%.c
        $(COMPILE.c) -o $@ $<
        $(POST_PROCESS_O)
```

```
**********************************************************
   42357 Sat Jun 18 19:37:59 2016
new/usr/src/lib/libdns_sd/java/common/JNISupport.c
7120 mDNS resync was not wsdiff safe
**********************************************************
______unchanged_portion_omitted_
1061 #endif


1064 // Note: The C preprocessor stringify operator ('#') makes a string from its arg
1065 // e.g. If "version" is #define'd to be "4", then STRINGIFY_AWE(version) will re
1066 // To expand "version" to its value before making the string, use STRINGIFY(vers
1067 #define STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s) # s
1068 #define STRINGIFY(s) STRINGIFY_ARGUMENT_WITHOUT_EXPANSION(s)

1070 // NOT static -- otherwise the compiler may optimize it out
1071 // The "@(#) " pattern is a special prefix the "what" command looks for
1072 #ifndef MDNS_VERSIONSTR_NODTS
1073 #endif /* ! codereview */
1074 const char VersionString_SCCS[] = "@(#) libjdns_sd " STRINGIFY(mDNSResponderVers
1075 #else
1076 const char VersionString_SCCS[] = "@(#) libjdns_sd " STRINGIFY(mDNSResponderVers
1077 #endif
1078 #endif /* ! codereview */
```

****
1242 Sat Jun 18 19:38:01 2016
new/usr/src/lib/libfru/amd64/Makefile
7069 i386_XARCH should include -m32
****

```
   1 #
   2 # CDDL HEADER START
   3 #
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License (the "License").
   6 # You may not use this file except in compliance with the License.
   7 #
   8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9 # or http://www.opensolaris.org/os/licensing.
  10 # See the License for the specific language governing permissions
  11 # and limitations under the License.
  12 #
  13 # When distributing Covered Code, include this CDDL HEADER in each
  14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15 # If applicable, add the following below this CDDL HEADER, with the
  16 # fields enclosed by brackets "[]" replaced with your own identifying
  17 # information: Portions Copyright [yyyy] [name of copyright owner]
  18 #
  19 # CDDL HEADER END
  20 #
  21 #
  22 # Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
  23 # Use is subject to license terms.
  24 #
  25 #

  27 # include common Makefiles
  28 include ../Makefile.obj
  29 include ../../Makefile.lib.64
  30 include ../Makefile.flag
  31 LDLIBS += -lc
  31 LDLIBS += -lc -m64
  32 LDLIBS += -L$(SRC)/lib/libfruutils/$(MACH64) -lfruutils
  33 LDLIBS += -L$(SRC)/lib/libfru/libfrureg/$(MACH64) -lfrureg

  35 install: all $(ROOTLIBDIR64) $(ROOTLINKSCCC64)

  37 $(ROOTLIBDIR64):
  38         $(INS.dir)

  40 include ../Makefile.targ
```

```
**********************************************************
    1272 Sat Jun 18 19:38:02 2016
new/usr/src/lib/libfru/sparcv9/Makefile
7069 i386_XARCH should include -m32
**********************************************************
```

```
   1 #
   2 # CDDL HEADER START
   3 #
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License (the "License").
   6 # You may not use this file except in compliance with the License.
   7 #
   8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9 # or http://www.opensolaris.org/os/licensing.
  10 # See the License for the specific language governing permissions
  11 # and limitations under the License.
  12 #
  13 # When distributing Covered Code, include this CDDL HEADER in each
  14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15 # If applicable, add the following below this CDDL HEADER, with the
  16 # fields enclosed by brackets "[]" replaced with your own identifying
  17 # information: Portions Copyright [yyyy] [name of copyright owner]
  18 #
  19 # CDDL HEADER END
  20 #
  21 #
  22 # Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
  23 # Use is subject to license terms.
  24 #
  25 # lib/libfru/sparcv9/Makefile
  26 #

  28 # include common Makefiles
  29 include ../Makefile.obj
  30 include ../../Makefile.lib.64
  31 include ../Makefile.flag
  32 LDLIBS += -lc
  32 LDLIBS += -lc -m64
  33 LDLIBS += -L$(SRC)/lib/libfruutils/$(MACH64) -lfruutils
  34 LDLIBS += -L$(SRC)/lib/libfru/libfrureg/$(MACH64) -lfrureg

  36 install: all $(ROOTLIBDIR64) $(ROOTLINKSCCC64)

  38 $(ROOTLIBDIR64):
  39         $(INS.dir)

  41 include ../Makefile.targ
```

```
**********************************************************
    1836 Sat Jun 18 19:38:03 2016
new/usr/src/test/libc-tests/tests/Makefile
7069 i386_XARCH should include -m32
**********************************************************
```

```
   1 #
   2 # This file and its contents are supplied under the terms of the
   3 # Common Development and Distribution License ("CDDL"), version 1.0.
   4 # You may only use this file in accordance with the terms of version
   5 # 1.0 of the CDDL.
   6 #
   7 # A full copy of the text of the CDDL should have accompanied this
   8 # source.  A copy of the CDDL is also available via the Internet at
   9 # http://www.illumos.org/license/CDDL.
  10 #

  12 #
  13 # Copyright (c) 2012 by Delphix. All rights reserved.
  14 # Copyright 2015 Garrett D'Amore <garrett@damore.org>
  15 # Copyright 2016 Joyent, Inc.
  16 #

  18 SUBDIRS = \
  19         catopen \
  20         fpround \
  21         newlocale \
  22         nl_langinfo \
  23         priv_gettext \
  24         random \
  25         strerror \
  26         symbols \
  27         threads \
  28         wcsrtombs \
  29         wctype

  31 PROGS = \
  32         aligned_alloc \
  33         c11_threads \
  34         c11_tss \
  35         call_once \
  36         endian \
  37         quick_exit_order \
  38         quick_exit_status \
  39         timespec_get

  41 SCRIPTS = \
  42         quick_exit

  44 CPPFLAGS += -D_REENTRANT

  46 PROGS32 = $(PROGS:%=%.32)
  47 PROGS64 = $(PROGS:%=%.64)

  49 aligned_alloc.32 :=     LDLIBS += -lproc
  50 aligned_alloc.64 :=     LDLIBS64 += -lproc

  52 ROOTOPTDIR = $(ROOT)/opt/libc-tests/tests
  53 ROOTOPTPROGS = $(PROGS32:%=$(ROOTOPTDIR)/%) \
  54         $(PROGS64:%=$(ROOTOPTDIR)/%) \
  55         $(SCRIPTS:%=$(ROOTOPTDIR)/%)

  57 include $(SRC)/cmd/Makefile.cmd

  59 all     :=      TARGET = all
  60 install :=      TARGET = install
  61 clean   :=      TARGET = clean
```



```
  62 clobber :=      TARGET = clobber
  63 lint    :=      TARGET = lint

  65 .KEEP_STATE:

  67 install: $(SUBDIRS) $(ROOTOPTPROGS)

  69 all: $(SUBDIRS) $(PROGS32) $(PROGS64)

  71 clean lint: $(SUBDIRS)

  73 $(ROOTOPTPROGS): $(PROGS32) $(PROGS64) $(ROOTOPTDIR)

  75 $(ROOTOPTDIR):
  76         $(INS.dir)

  78 $(ROOTOPTDIR)/%: %
  79         $(INS.file)

  81 $(ROOTOPTDIR)/%: %.ksh
  82         $(INS.rename)

  84 %.64: %.c
  85         $(LINK64.c) -o $@ $< $(LDLIBS64)
  86         $(POST_PROCESS)

  88 %.32: %.c
  89         $(LINK.c) -o $@ $< $(LDLIBS)
  89         $(LINK.c) -m32 -o $@ $< $(LDLIBS)
  90         $(POST_PROCESS)

  92 clobber: $(SUBDIRS)
  93         $(RM) $(PROGS32) $(PROGS64)

  95 $(SUBDIRS): FRC
  96         @cd $@; pwd; $(MAKE) $(TARGET)

  98 FRC:
```