

new/usr/src/Makefile.master

1

```
*****
35131 Sun Sep 6 19:01:39 2015
new/usr/src/Makefile.master
6117 Many small bugs prevent a clean build on SPARC
*****
1 #
2 # CDDL HEADER START
3 #
4 # The contents of this file are subject to the terms of the
5 # Common Development and Distribution License (the "License").
6 # You may not use this file except in compliance with the License.
7 #
8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9 # or http://www.opensolaris.org/os/licensing.
10 # See the License for the specific language governing permissions
11 # and limitations under the License.
12 #
13 # When distributing Covered Code, include this CDDL HEADER in each
14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 # If applicable, add the following below this CDDL HEADER, with the
16 # fields enclosed by brackets "[]" replaced with your own identifying
17 # information: Portions Copyright [yyyy] [name of copyright owner]
18 #
19 # CDDL HEADER END
20 #
21 #
22 #
23 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright (c) 2012 by Delphix. All rights reserved.
25 # Copyright 2014 Garrett D'Amore <garrett@damore.org>
26 # Copyright 2015, OmniTI Computer Consulting, Inc. All rights reserved.
27 # Copyright 2015 Gary Mills
28 #
29 #
30 #
31 # Makefile.master, global definitions for system source
32 #
33 ROOT=          /proto
34 #
35 #
36 # Adjunct root, containing an additional proto area to be used for headers
37 # and libraries.
38 #
39 ADJUNCT_PROTO=
40 #
41 #
42 # Adjunct for building things that run on the build machine.
43 #
44 NATIVE_ADJUNCT= /usr
45 #
46 #
47 # RELEASE_BUILD should be cleared for final release builds.
48 # NOT_RELEASE_BUILD is exactly what the name implies.
49 #
50 # __GNUC toggles the building of ON components using gcc and related tools.
51 # Normally set to '#', set it to '' to do gcc build.
52 #
53 # The declaration POUND_SIGN is always '#'. This is needed to get around the
54 # make feature that '#' is always a comment delimiter, even when escaped or
55 # quoted. We use this macro expansion method to get POUND_SIGN rather than
56 # always breaking out a shell because the general case can cause a noticeable
57 # slowdown in build times when so many Makefiles include Makefile.master.
58 #
59 # While the majority of users are expected to override the setting below
60 # with an env file (via nightly or bldenv), if you aren't building that way
61 # (ie, you're using "ws" or some other bootstrapping method) then you need
```

new/usr/src/Makefile.master

2

```
62 # this definition in order to avoid the subshell invocation mentioned above.
63 #
64 #
65 PRE_POUND=          pre\#
66 POUND_SIGN=         $(PRE_POUND:pre\#=%%)
67 #
68 NOT_RELEASE_BUILD=
69 RELEASE_BUILD=      $(POUND_SIGN)
70 $(RELEASE_BUILD)NOT_RELEASE_BUILD= $(POUND_SIGN)
71 PATCH_BUILD=        $(POUND_SIGN)
72 #
73 # SPARC_BLD is '#' for an Intel build.
74 # INTEL_BLD is '#' for a Sparc build.
75 SPARC_BLD_1=        $(MACH:i386=$(POUND_SIGN))
76 SPARC_BLD=          $(SPARC_BLD_1:sparc=)
77 INTEL_BLD_1=        $(MACH:sparc=$(POUND_SIGN))
78 INTEL_BLD=          $(INTEL_BLD_1:i386=)
79 #
80 # The variables below control the compilers used during the build.
81 # There are a number of permutations.
82 #
83 # __GNUC and __SUNC control (and indicate) the primary compiler. Whichever
84 # one is not POUND_SIGN is the primary, with the other as the shadow. They
85 # may also be used to control entirely compiler-specific Makefile assignments.
86 # __GNUC and GCC are the default.
87 #
88 # __GNUC64 indicates that the 64bit build should use the GNU C compiler.
89 # There is no Sun C analogue.
90 #
91 # The following version-specific options are operative regardless of which
92 # compiler is primary, and control the versions of the given compilers to be
93 # used. They also allow compiler-version specific Makefile fragments.
94 #
95 #
96 __SUNC=              $(POUND_SIGN)
97 $(__SUNC)__GNUC=     $(POUND_SIGN)
98 __GNUC64=           $(__GNUC)
99 #
100 # Allow build-time "configuration" to enable or disable some things.
101 # The default is POUND_SIGN, meaning "not enabled". If the environment
102 # passes in an override like ENABLE_SMB_PRINTING= (empty) that will
103 # uncomment things in the lower Makefiles to enable the feature.
104 ENABLE_IPP_PRINTING= $(POUND_SIGN)
105 ENABLE_SMB_PRINTING= $(POUND_SIGN)
106 #
107 # CLOSED is the root of the tree that contains source which isn't released
108 # as open source
109 CLOSED=              $(SRC)/../closed
110 #
111 # BUILD_TOOLS is the root of all tools including compilers.
112 # ONBLD_TOOLS is the root of all the tools that are part of SUNWonbld.
113 #
114 BUILD_TOOLS=         /ws/onnv-tools
115 ONBLD_TOOLS=         $(BUILD_TOOLS)/onbld
116 #
117 # define runtime JAVA_HOME, primarily for cmd/pools/poold
118 JAVA_HOME=           /usr/java
119 # define buildtime JAVA_ROOT
120 JAVA_ROOT=           /usr/java
121 #
122 GCC_ROOT=            /opt/gcc/4.4.4
123 GCCLIBDIR=           $(GCC_ROOT)/lib
124 GCCLIBDIR64=         $(GCC_ROOT)/lib/$(MACH64)
125 #
126 DOCBOOK_XSL_ROOT=   /usr/share/sgml/docbook/xsl-stylesheets
```

```

128 RPCGEN=      /usr/bin/rpcgen
129 STABS=       $(ONBLD_TOOLS)/bin/$(MACH)/stabs
130 ELFXTRACT=   $(ONBLD_TOOLS)/bin/$(MACH)/elfextract
131 MBH_PATCH=   $(ONBLD_TOOLS)/bin/$(MACH)/mbh_patch
132 ECHO=        echo
133 INS=         install
134 TRUE=        true
135 SYMLINK=     /usr/bin/ln -s
136 LN=         /usr/bin/ln
137 CHMOD=      /usr/bin/chmod
138 MV=         /usr/bin/mv -f
139 RM=         /usr/bin/rm -f
140 CUT=        /usr/bin/cut
141 NM=         /usr/ccs/bin/nm
142 DIFF=       /usr/bin/diff
143 GREP=       /usr/bin/grep
144 EGREP=      /usr/bin/egrep
145 ELFWRAP=    /usr/bin/elfwrap
146 KSH93=     /usr/bin/ksh93
147 SED=       /usr/bin/sed
148 NAWK=      /usr/bin/nawk
149 CP=        /usr/bin/cp -f
150 MCS=       /usr/ccs/bin/mcs
151 CAT=       /usr/bin/cat
152 ELFDUMP=   /usr/ccs/bin/elfdump
153 M4=        /usr/bin/m4
154 M4=        /usr/ccs/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 ELFDIT=   /usr/bin/elfedit
185 ELFSIGN=  /usr/bin/elfsign
186 DTRACE=   /usr/sbin/dtrace -xnolib
187 UNIQ=     /usr/bin/uniq
188 TAR=      /usr/bin/tar
189 ASTBINDIR= /usr/ast/bin
190 MSGCC=    $(ASTBINDIR)/msgcc

191 FILEMODE= 644
192 DIRMODE=  755

```

```

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

198 # For stylistic checks
199 #
200 # Note that the X and C checks are not used at this time and may need
201 # modification when they are actually used.
202 #
203 CSTYLE=      $(ONBLD_TOOLS)/bin/cstyle
204 CSTYLE_TAIL=
205 HDRCHK=      $(ONBLD_TOOLS)/bin/hdrchk
206 HDRCHK_TAIL=
207 JSTYLE=      $(ONBLD_TOOLS)/bin/jstyle

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

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

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

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

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

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

238 #
239 # Python bakes the mtime of the .py file into the compiled .pyc and
240 # rebuilds if the baked-in mtime != the mtime of the source file
241 # (rather than only if it's less than), thus when installing python
242 # files we must make certain to not adjust the mtime of the source
243 # (.py) file.
244 #
245 INS.pyfile=   $(INS.file); $(TOUCH) -r $< $@

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

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

```

```

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

263 sparc_BUILD64=
264 i386_BUILD64=
265 BUILD64=      $$($(MACH)_BUILD64)

267 #
268 # C compiler mode. Future compilers may change the default on us,
269 # so force extended ANSI mode globally. Lower level makefiles can
270 # override this by setting CCMODE.
271 #
272 CCMODE=         -Xa
273 CCMODE64=      -Xa

275 #
276 # C compiler verbose mode. This is so we can enable it globally,
277 # but turn it off in the lower level makefiles of things we cannot
278 # (or aren't going to) fix.
279 #
280 CCVERBOSE=     -v

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

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

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

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

317 # One optimization the compiler might perform is to turn this:
318 #   #pragma weak foo
319 #   extern int foo;
320 #   if (&foo)
321 #       foo = 5;
322 # into
323 #   foo = 5;
324 # Since we do some of this (foo might be referenced in common kernel code

```

```

325 # but provided only for some cpu modules or platforms), we disable this
326 # optimization.
327 #
328 sparc_CCUNBOUND = -Wd,-xsafe=unboundsym
329 i386_CCUNBOUND =
330 CCUNBOUND =     $$($(MACH)_CCUNBOUND)

332 #
333 # compiler '-xarch' flag. This is here to centralize it and make it
334 # overridable for testing.
335 sparc_XARCH=    -m32
336 sparcv9_XARCH= -m64
337 i386_XARCH=
338 amd64_XARCH=   -m64 -Ui386 -U__i386

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

346 #
347 # These flags define what we need to be 'standalone' i.e. -not- part
348 # of the rather more cosy userland environment. This basically means
349 # the kernel.
350 #
351 # XX64 future versions of gcc will make -mmodel=kernel imply -mno-red-zone
352 #
353 sparc_STAND_FLAGS=  _gcc=-ffreestanding
354 sparcv9_STAND_FLAGS= _gcc=-ffreestanding
355 # Disabling MMX also disables 3DNow, disabling SSE also disables all later
356 # additions to SSE (SSE2, AVX ,etc.)
357 NO_SIMD=           _gcc=-mno-mmx _gcc=-mno-sse
358 i386_STAND_FLAGS=  _gcc=-ffreestanding $(NO_SIMD)
359 amd64_STAND_FLAGS= -xmodel=kernel $(NO_SIMD)

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

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

367 #
368 # disable the incremental linker
369 ILDOFF=           -xildoff
370 #
371 XDEPEND=          -xdepend
372 XFFLAG=           -xF=%all
373 XESS=             -xs
374 XSTRCONST=       -xstrconst

376 #
377 # turn warnings into errors (C)
378 CERRWARN = -errtags=yes -errwarn=%all
379 CERRWARN += -erroff=E_EMPTY_TRANSLATION_UNIT
380 CERRWARN += -erroff=E_STATEMENT_NOT_REACHED

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

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

```

```

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

396 #
397 # turn warnings into errors (C++)
398 CCERRWARN= -xwe

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

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

414 sparc_ASFLAGS= $(sparc_AS_XARCH)
415 sparcv9_ASFLAGS=$(sparcv9_AS_XARCH)
416 i386_ASFLAGS= $(i386_AS_XARCH)
417 amd64_ASFLAGS= $(amd64_AS_XARCH)

419 #
420 sparc_COPTFLAG= -xO3
421 sparcv9_COPTFLAG= -xO3
422 i386_COPTFLAG= -O
423 amd64_COPTFLAG= -xO3

425 COPTFLAG= $($(_MACH)_COPTFLAG)
426 COPTFLAG64= $($(_MACH64)_COPTFLAG)

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

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

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

444 #
445 # Default debug format for Sun Studio 11 is dwarf, so force it to
446 # generate stabs.
447 #
448 DEBUGFORMAT= -xdebugformat=stabs

450 #
451 # Flags used to build in debug mode for ctf generation. Bugs in the Devpro
452 # compilers currently prevent us from building with cc-emitted DWARF.
453 #
454 CTF_FLAGS_sparc = -g -Wc,-Qiselect-T1 $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)
455 CTF_FLAGS_i386 = -g $(C99MODE) $(CNOGLOBAL) $(CDWARFSTR)

```

```

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

460 # Sun Studio produces broken userland code when saving arguments.
461 $(__GNUCC)CTF_FLAGS_amd64 += $(SAVEARGS)

463 CTF_FLAGS_32 = $(CTF_FLAGS_$_MACH) $(DEBUGFORMAT)
464 CTF_FLAGS_64 = $(CTF_FLAGS_$_MACH64) $(DEBUGFORMAT)
465 CTF_FLAGS = $(CTF_FLAGS_32)

467 #
468 # Flags used with genoffsets
469 #
470 GOFLAGS = -_noecho \
471 $(CALLSYMS) \
472 $(CDWARFSTR)

474 OFFSETS_CREATE = $(GENOFFSETS) -s $(CTFSTABS) -r $(CTFCONVERT) \
475 $(CC) $(GOFLAGS) $(CFLAGS) $(CPPFLAGS)

477 OFFSETS_CREATE64 = $(GENOFFSETS) -s $(CTFSTABS) -r $(CTFCONVERT) \
478 $(CC) $(GOFLAGS) $(CFLAGS64) $(CPPFLAGS)

480 #
481 # tradeoff time for space (smaller is better)
482 #
483 sparc_SPACEFLAG = -xspace -W0,-Lt
484 sparcv9_SPACEFLAG = -xspace -W0,-Lt
485 i386_SPACEFLAG = -xspace
486 amd64_SPACEFLAG =

488 SPACEFLAG = $($(_MACH)_SPACEFLAG)
489 SPACEFLAG64 = $($(_MACH64)_SPACEFLAG)

491 #
492 # The Sun Studio 11 compiler has changed the behaviour of integer
493 # wrap arounds and so a flag is needed to use the legacy behaviour
494 # (without this flag panics/hangs could be exposed within the source).
495 #
496 sparc_IROPTFLAG = -W2,-xwrap_int
497 sparcv9_IROPTFLAG = -W2,-xwrap_int
498 i386_IROPTFLAG =
499 amd64_IROPTFLAG =

501 IROPTFLAG = $($(_MACH)_IROPTFLAG)
502 IROPTFLAG64 = $($(_MACH64)_IROPTFLAG)

504 sparc_XREGSFLAG = -xregs=no%appl
505 sparcv9_XREGSFLAG = -xregs=no%appl
506 i386_XREGSFLAG =
507 amd64_XREGSFLAG =

509 XREGSFLAG = $($(_MACH)_XREGSFLAG)
510 XREGSFLAG64 = $($(_MACH64)_XREGSFLAG)

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

517 #
518 # These variables are intended ONLY for use by developers to safely pass extra
519 # flags to the compilers without unintentionally overriding Makefile-set
520 # flags. They should NEVER be set to any value in a Makefile.
521 #
522 # They come last in the associated FLAGS variable such that they can

```

```

523 # explicitly override things if necessary, there are gaps in this, but it's
524 # the best we can manage.
525 #
526 CUSERFLAGS           =
527 CUSERFLAGS64         = $(CUSERFLAGS)
528 CCUSERFLAGS          =
529 CCUSERFLAGS64        = $(CCUSERFLAGS)

531 CSOURCEDEBUGFLAGS    =
532 CCSOURCEDEBUGFLAGS   =
533 $(SRCDGBLD)CSOURCEDEBUGFLAGS = -g -xs
534 $(SRCDGBLD)CCSOURCEDEBUGFLAGS = -g -xs

536 CFLAGS=              $(COPTFLAG) $(MACH)_CFLAGS $(SPACEFLAG) $(CCMODE) \
537 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG) \
538 $(GLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
539 $(CUSERFLAGS)
540 CFLAGS64=            $(COPTFLAG64) $(MACH64)_CFLAGS $(SPACEFLAG64) $(CCMODE64) \
541 $(ILDOFF) $(CERRWARN) $(C99MODE) $(CCUNBOUND) $(IROPTFLAG64) \
542 $(GLOBALSTATIC) $(CCNOAUTOINLINE) $(CSOURCEDEBUGFLAGS) \
543 $(CUSERFLAGS64)
544 #
545 # Flags that are used to build parts of the code that are subsequently
546 # run on the build machine (also known as the NATIVE_BUILD).
547 #
548 NATIVE_CFLAGS=       $(COPTFLAG) $(NATIVE_MACH)_CFLAGS $(CCMODE) \
549 $(ILDOFF) $(CERRWARN) $(C99MODE) $(NATIVE_MACH)_CCUNBOUND) \
550 $(IROPTFLAG) $(GLOBALSTATIC) $(CCNOAUTOINLINE) \
551 $(CSOURCEDEBUGFLAGS) $(CUSERFLAGS)

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

565 #
566 # For source message catalogue
567 #
568 .SUFFIXES: $(SUFFIXES) .i .po
569 MSGROOT= $(ROOT)/catalog
570 MSGDOMAIN= $(MSGROOT)/$(TEXT_DOMAIN)
571 MSGDOMAINPOFILE = $(MSGDOMAIN)/$(POFILE)
572 DCMMSGDOMAIN= $(MSGROOT)/LC_TIME/$(TEXT_DOMAIN)
573 DCMMSGDOMAINPOFILE = $(DCMSGDOMAIN)/$(DCFILE:.dc=.po)

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

587 #
588 # This is overwritten by local Makefile when PROG is a list.

```

```

589 #
590 POFILE= $(PROG).po

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

609 sparc_CCOPTFLAG=     -O
610 sparcv9_CCOPTFLAG=   -O
611 i386_CCOPTFLAG=      -O
612 amd64_CCOPTFLAG=     -O

614 CCOPTFLAG=           $(MACH)_CCOPTFLAG)
615 CCOPTFLAG64=         $(MACH64)_CCOPTFLAG)
616 CCFLAGS=             $(CCOPTFLAG) $(MACH)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
617 $(CUSERFLAGS)
618 CCFLAGS64=           $(CCOPTFLAG64) $(MACH64)_CCFLAGS) $(CCSOURCEDEBUGFLAGS) \
619 $(CUSERFLAGS64)

621 #
622 #
623 #
624 ELFWRAP_FLAGS =
625 ELFWRAP_FLAGS64 = -64

627 #
628 # Various mapfiles that are used throughout the build, and delivered to
629 # /usr/lib/ld.
630 #
631 MAPFILE.NED_i386 = $(SRC)/common/mapfiles/common/map.noexdata
632 MAPFILE.NED_sparc =
633 MAPFILE.NED = $(MAPFILE.NED_$(MACH))
634 MAPFILE.PGA = $(SRC)/common/mapfiles/common/map.pagealign
635 MAPFILE.NES = $(SRC)/common/mapfiles/common/map.noexstk
636 MAPFILE.FLT = $(SRC)/common/mapfiles/common/map.filter
637 MAPFILE.LEX = $(SRC)/common/mapfiles/common/map.lex.yy

639 #
640 # Generated mapfiles that are compiler specific, and used throughout the
641 # build. These mapfiles are not delivered in /usr/lib/ld.
642 #
643 MAPFILE.NGB_sparc= $(SRC)/common/mapfiles/gen/sparc_cc_map.noexglobs
644 $(__GNUCC64)MAPFILE.NGB_sparc= \
645 $(SRC)/common/mapfiles/gen/sparc_gcc_map.noexglobs
646 MAPFILE.NGB_sparcv9= $(SRC)/common/mapfiles/gen/sparcv9_cc_map.noexglobs
647 $(__GNUCC64)MAPFILE.NGB_sparcv9= \
648 $(SRC)/common/mapfiles/gen/sparcv9_gcc_map.noexglobs
649 MAPFILE.NGB_i386= $(SRC)/common/mapfiles/gen/i386_cc_map.noexglobs
650 $(__GNUCC64)MAPFILE.NGB_i386= \
651 $(SRC)/common/mapfiles/gen/i386_gcc_map.noexglobs
652 MAPFILE.NGB_amd64= $(SRC)/common/mapfiles/gen/amd64_cc_map.noexglobs
653 $(__GNUCC64)MAPFILE.NGB_amd64= \
654 $(SRC)/common/mapfiles/gen/amd64_gcc_map.noexglobs

```

```

655 MAPFILE.NGB =          $(MAPFILE.NGB_$(MACH))

657 #
658 # A generic interface mapfile name, used by various dynamic objects to define
659 # the interfaces and interposers the object must export.
660 #
661 MAPFILE.INT =          mapfile-intf

663 #
664 # LDLIBS32 and LDLIBS64 can be set in the environment to override the following
665 # assignments.
666 #
667 # These environment settings make sure that no libraries are searched outside
668 # of the local workspace proto area:
669 #     LDLIBS32=-YP,$ROOT/lib:$ROOT/usr/lib
670 #     LDLIBS64=-YP,$ROOT/lib:$MACH64:$ROOT/usr/lib:$MACH64
671 #
672 LDLIBS32 =          $(ENVLDLIBS1) $(ENVLDLIBS2) $(ENVLDLIBS3)
673 LDLIBS32 +=        $(ADJUNCT_PROTO:%=-L%/usr/lib -L%/lib)
674 LDLIBS.cmd =       $(LDLIBS32)
675 LDLIBS.lib =       $(LDLIBS32)

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

682 #
683 # Define compilation macros.
684 #
685 COMPILER.c=         $(CC) $(CFLAGS) $(CPPFLAGS) -c
686 COMPILER64.c=       $(CC) $(CFLAGS64) $(CPPFLAGS) -c
687 COMPILER.cc=        $(CCC) $(CCFLAGS) $(CPPFLAGS) -c
688 COMPILER64.cc=      $(CCC) $(CCFLAGS64) $(CPPFLAGS) -c
689 COMPILER.s=         $(AS) $(ASFLAGS) $(AS_CPPFLAGS)
690 COMPILER64.s=       $(AS) $(ASFLAGS) $(MACH64)_AS_XARCH $(AS_CPPFLAGS)
691 COMPILER.d=         $(DTRACE) -G -32
692 COMPILER64.d=       $(DTRACE) -G -64
693 COMPILER.b=         $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))
694 COMPILER64.b=       $(ELFWRAP) $(ELFWRAP_FLAGS$(CLASS))

696 CLASSPATH=
697 COMPILER.java=     $(JAVAC) $(JAVAFLAGS) -classpath $(CLASSPATH)

699 #
700 # Link time macros
701 #
702 CCNEEDED            = -lc
703 CCEXTNEEDED        = -lcrun -lcstd
704 $(__GNUC)CCNEEDED  = -L$(GCCLIBDIR) -lstl++ -lgcc_s
705 $(__GNUC)CCEXTNEEDED = $(CCNEEDED)

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

715 #
716 # lint macros
717 #
718 # Note that the undefine of __PRAGMA_REDEFINE_EXTNAME can be removed once
719 # ON is built with a version of lint that has the fix for 4484186.
720 #

```

```

721 ALWAYS_LINT_DEFS = -errtags=yes -s
722 ALWAYS_LINT_DEFS += -erroff=E_PTRDIFF_OVERFLOW
723 ALWAYS_LINT_DEFS += -erroff=E_ASSIGN_NARROW_CONV
724 ALWAYS_LINT_DEFS += -U__PRAGMA_REDEFINE_EXTNAME
725 ALWAYS_LINT_DEFS += $(C99LMODE)
726 ALWAYS_LINT_DEFS += -errsecurity=$(SECLEVEL)
727 ALWAYS_LINT_DEFS += -erroff=E_SEC_CREAT_WITHOUT_EXCL
728 ALWAYS_LINT_DEFS += -erroff=E_SEC_FORBIDDEN_WARN_CREAT
729 # XX64 -- really only needed for amd64 lint
730 ALWAYS_LINT_DEFS += -erroff=E_ASSIGN_INT_TO_SMALL_INT
731 ALWAYS_LINT_DEFS += -erroff=E_CAST_INT_CONST_TO_SMALL_INT
732 ALWAYS_LINT_DEFS += -erroff=E_CAST_INT_TO_SMALL_INT
733 ALWAYS_LINT_DEFS += -erroff=E_CAST_TO_PTR_FROM_INT
734 ALWAYS_LINT_DEFS += -erroff=E_COMP_INT_WITH_LARGE_INT
735 ALWAYS_LINT_DEFS += -erroff=E_INTEGRAL_CONST_EXP_EXPECTED
736 ALWAYS_LINT_DEFS += -erroff=E_PASS_INT_TO_SMALL_INT
737 ALWAYS_LINT_DEFS += -erroff=E_PTR_CONV_LOSES_BITS

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

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

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

757 # Define native compilation macros
758 #

760 # Base directory where compilers are loaded.
761 # Defined here so it can be overridden by developer.
762 #
763 SPRO_ROOT=         $(BUILD_TOOLS)/SUNWspr0
764 SPRO_VROOT=        $(SPRO_ROOT)/SS12
765 GNU_ROOT=          /usr

767 # Till SS12ul formally becomes the NV CBE, LINT is hard
768 # coded to be picked up from the $SPRO_ROOT/sunstudio12.1/
769 # location. Impacted variables are sparc_LINT, sparcv9_LINT,
770 # i386_LINT, amd64_LINT.
771 # Reset them when SS12ul is rolled out.
772 #

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

786 sparcv9_CC=        $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc

```

```

787 $(__GNUC64)sparcv9_CC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
788 sparcv9_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
789 $(__GNUC64)sparcv9_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
790 sparcv9_CPP= /usr/ccs/lib/cpp
791 sparcv9_AS= /usr/ccs/bin/as -xregsym=no
792 sparcv9_LD= /usr/ccs/bin/ld
793 sparcv9_LINT= $(SPRO_ROOT)/sunstudio12.1/bin/lint

795 i386_CC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_cc
796 $(__GNUC)i386_CC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_gcc
797 i386_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_CC
798 $(__GNUC)i386_CCC= $(ONBLD_TOOLS)/bin/$(MACH)/cw -_g++
799 i386_CPP= /usr/ccs/lib/cpp
800 i386_AS= /usr/ccs/bin/as
801 $(__GNUC)i386_AS= $(ONBLD_TOOLS)/bin/$(MACH)/aw
802 i386_LD= /usr/ccs/bin/ld
803 i386_LINT= $(SPRO_ROOT)/sunstudio12.1/bin/lint

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

814 NATIVECC= $($ (NATIVE_MACH)_CC)
815 NATIVECCC= $($ (NATIVE_MACH)_CCC)
816 NATIVECPP= $($ (NATIVE_MACH)_CPP)
817 NATIVEAS= $($ (NATIVE_MACH)_AS)
818 NATVELD= $($ (NATIVE_MACH)_LD)
819 NATVELINT= $($ (NATIVE_MACH)_LINT)

821 #
822 # Makefile.master.64 overrides these settings
823 #
824 CC= $(NATIVECC)
825 CCC= $(NATIVECCC)
826 CPP= $(NATIVECPP)
827 AS= $(NATIVEAS)
828 LD= $(NATVELD)
829 LINT= $(NATVELINT)

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

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

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

847 ZDEFS= -zdefs
848 ZDIRECT= -zdirect
849 ZIGNORE= -zignore
850 ZINITFIRST= -zinitfirst
851 ZINTERPOSE= -zinterpose
852 ZLAZYLOAD= -zlazyload

```

```

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

869 GSHARED= -G
870 CCMT= -mt

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

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

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

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

897 AS_PICFLAGS= $(C_PICFLAGS)
898 AS_BIGPICFLAGS= $(C_BIGPICFLAGS)

900 #
901 # Default label for CTF sections
902 #
903 CTFCVTFLAGS= -i -L VERSION

905 #
906 # Override to pass module-specific flags to ctmerge. Currently used only by
907 # krtld to turn on fuzzy matching, and source-level debugging to inhibit
908 # stripping.
909 #
910 CTFMRGFLAGS=

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

914 ELFSIGN_O= $(TRUE)
915 ELFSIGN_CRYPT= $(ELFSIGN_O)
916 ELFSIGN_OBJECT= $(ELFSIGN_O)

918 # Rules (normally from make.rules) and macros which are used for post

```

```

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

970 PROCESS_COMMENT= @?${MCS} -d -a $(RELEASE_CM) -a $(DEV_CM)
971 $(RELEASE_BUILD)PROCESS_COMMENT= @?${MCS} -d -a $(RELEASE_CM)

973 STRIP_STABS=      $(STRIP) -x $@
974 $(SRCDBGBLD)STRIP_STABS=
:

976 POST_PROCESS_O=
977 POST_PROCESS_A=
978 POST_PROCESS_SO= $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
979                  $(ELFSIGN_OBJECT)
980 POST_PROCESS=     $(PROCESS_COMMENT) $@ ; $(STRIP_STABS) ; \
981                  $(ELFSIGN_OBJECT)

983 #
984 # chk4ubin is a tool that inspects a module for a symbol table

```

```

985 # ELF section size which can trigger an OBP bug on older platforms.
986 # This problem affects only specific sun4u bootable modules.
987 #
988 CHK4UBIN=      $(ONBLD_TOOLS)/bin/$(MACH)/chk4ubin
989 CHK4UBINFLAGS=
990 CHK4UBINARY=   $(CHK4UBIN) $(CHK4UBINFLAGS) $@

992 #
993 # PKGARCHIVE specifies the default location where packages should be
994 # placed if built.
995 #
996 $(RELEASE_BUILD)PKGARCHIVESUFFIX= -nd
997 PKGARCHIVE=$(SRC)/../../packages/$(MACH)/nightly$(PKGARCHIVESUFFIX)

999 #
1000 # The repositories will be created with these publisher settings. To
1001 # update an image to the resulting repositories, this must match the
1002 # publisher name provided to "pkg set-publisher."
1003 #
1004 PKGPUBLISHER_REDIST= on-nightly
1005 PKGPUBLISHER_NONREDIST= on-extra

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

```



```

1051      $(LEX.l) $< > $*.c
1052      $(LINK.c) -o $@ $*.c -ll $(LDLIBS)
1053      $(POST_PROCESS)
1054      $(RM) $*.c
1055 .l.o:
1056      $(RM) $*.c
1057      $(LEX.l) $< > $*.c
1058      $(COMPILE.c) -o $@ $*.c $(CTFCONVERT_HOOK)
1059      $(POST_PROCESS_O)
1060      $(RM) $*.c

1062 .bin.o:
1063      $(COMPILE.b) -o $@ $<
1064      $(POST_PROCESS_O)

1066 .java.class:
1067      $(COMPILE.java) $<

1069 # Bourne and Korn shell script message catalog build rules.
1070 # We extract all gettext strings with sed(1) (being careful to permit
1071 # multiple gettext strings on the same line), weed out the dups, and
1072 # build the catalogue with awk(1).

1074 .sh.po .ksh.po:
1075      $(SED) -n -e ":a" \
1076      -e "h" \
1077      -e "s/.*gettext *\([^\"]*\)\.*/\1/p" \
1078      -e "x" \
1079      -e "s/\(.*\)gettext *\"[^\"]*\(.*\)\/\1\2/" \
1080      -e "t a" \
1081      $< | sort -u | awk '{ print "msgid\t" $$0 "\nmsgstr" }' > $@

1083 #
1084 # Python and Perl executable and message catalog build rules.
1085 #
1086 .SUFFIXES: .pl .pm .py .pyc

1088 .pl:
1089      $(RM) $@;
1090      $(SED) -e "s@TEXT_DOMAIN@\"$(TEXT_DOMAIN)\@" $< > $@;
1091      $(CHMOD) +x $@

1093 .py:
1094      $(RM) $@; $(CAT) $< > $@; $(CHMOD) +x $@

1096 .py.pyc:
1097      $(RM) $@
1098      $(PYTHON) -mpy_compile $<
1099      @[ $(<)c = $@ ] || $(MV) $(<)c $@

1101 .py.po:
1102      $(GNUXGETTEXT) $(GNUXGETFLAGS) -d $(<F:%.py=%) $< ;

1104 .pl.po .pm.po:
1105      $(XGETTEXT) $(XGETFLAGS) -d $(<F) $< ;
1106      $(RM) $@ ;
1107      $(SED) "/^domain/d" < $(<F).po > $@ ;
1108      $(RM) $(<F).po

1110 #
1111 # When using xgettext, we want messages to go to the default domain,
1112 # rather than the specified one. This special version of the
1113 # COMPILE.cpp macro effectively prevents expansion of TEXT_DOMAIN,
1114 # causing xgettext to put all messages into the default domain.
1115 #
1116 CPPFORPO=$(COMPILE.cpp:\ "$(TEXT_DOMAIN)"=TEXT_DOMAIN)

```

```

1118 .c.i:
1119      $(CPPFORPO) $< > $@

1121 .h.i:
1122      $(CPPFORPO) $< > $@

1124 .y.i:
1125      $(YACC) -d $<
1126      $(CPPFORPO) y.tab.c > $@
1127      $(RM) y.tab.c

1129 .l.i:
1130      $(LEX) $<
1131      $(CPPFORPO) lex.yy.c > $@
1132      $(RM) lex.yy.c

1134 .c.po:
1135      $(CPPFORPO) $< > $<.i
1136      $(BUILD.po)

1138 .cc.po:
1139      $(CPPFORPO) $< > $<.i
1140      $(BUILD.po)

1142 .y.po:
1143      $(YACC) -d $<
1144      $(CPPFORPO) y.tab.c > $<.i
1145      $(BUILD.po)
1146      $(RM) y.tab.c

1148 .l.po:
1149      $(LEX) $<
1150      $(CPPFORPO) lex.yy.c > $<.i
1151      $(BUILD.po)
1152      $(RM) lex.yy.c

1154 #
1155 # Rules to perform stylistic checks
1156 #
1157 .SUFFIXES: .x .xml .check .xmlchk

1159 .h.check:
1160      $(DOT_H_CHECK)

1162 .x.check:
1163      $(DOT_X_CHECK)

1165 .xml.xmlchk:
1166      $(MANIFEST_CHECK)

1168 #
1169 # Include rules to render automated sccs get rules "safe".
1170 #
1171 include $(SRC)/Makefile.noget

```

new/usr/src/cmd/mdb/sparc/v7/libfksmbsrv/Makefile

1

1658 Sun Sep 6 19:01:39 2015

new/usr/src/cmd/mdb/sparc/v7/libfksmbsrv/Makefile

6117 Many small bugs prevent a clean build on SPARC

```
1 #
2 # CDDL HEADER START
3 #
4 # The contents of this file are subject to the terms of the
5 # Common Development and Distribution License, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 #
23 # Copyright 2005 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
26 # Copyright 2013 Nexenta Systems, Inc. All rights reserved.
27 # Copyright 2015 Gary Mills
28 #
29
30 MODULE = libfksmbsrv.so
31 MDBTGT = proc
32
33 MODSRCS = smbsrv.c list.c
34
35 include ../../../../Makefile.cmd
36 include ../../Makefile.sparcv7
37 include ../../Makefile.module
38
39 MODSRCS_DIR = ../../../../common/modules/smbsrv
40 GENUNIX_DIR = ../../../../common/modules/genunix
41
42 # Note: need our sys includes _before_ ENVCPPFLAGS, proto etc.
43 CPPFLAGS.first += -I../../../../../../lib/smbsrv/libfksmbsrv/common
44 CPPFLAGS.first += -I../../../../../../lib/libfakekernel/common
45
46 CPPFLAGS += -I../../../../../../uts/common
47
48 C99MODE=      -xc99=%all
49 C99LMODE=     -Xc99=%all
50
51 dmod%.o: $(GENUNIX_DIR)/%.c
52         $(COMPILE.c) -o $@ $<
53         $(CTFCONVERT_O)
54
55 dmod%.ln: $(GENUNIX_DIR)/%.c
56         $(LINT.c) -c $<
```

new/usr/src/cmd/mdb/sparc/v9/libfksmbsrv/Makefile

1

1694 Sun Sep 6 19:01:39 2015

new/usr/src/cmd/mdb/sparc/v9/libfksmbsrv/Makefile

6117 Many small bugs prevent a clean build on SPARC

```
1 #
2 # CDDL HEADER START
3 #
4 # The contents of this file are subject to the terms of the
5 # Common Development and Distribution License, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 #
23 # Copyright 2005 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
26 # Copyright 2013 Nexenta Systems, Inc. All rights reserved.
27 # Copyright 2015 Gary Mills
28 #

30 MODULE = libfksmbsrv.so
31 MDBGT = proc

33 MODSRCS = smbsrv.c list.c

35 include ../../../../Makefile.cmd
36 include ../../../../Makefile.cmd.64
37 include ../../Makefile.sparcv9
38 include ../../../../Makefile.module

40 MODSRCS_DIR = ../../../../common/modules/smbsrv
41 GENUNIX_DIR = ../../../../common/modules/genunix

43 # Note: need our sys includes _before_ ENVCPPFLAGS, proto etc.
44 CPPFLAGS.first += -I../../../../lib/smbsrv/libfksmbsrv/common
45 CPPFLAGS.first += -I../../../../lib/libfakekernel/common

47 CPPFLAGS += -I../../../../uts/common

49 C99MODE= -xc99=%all
50 C99LMODE= -Xc99=%all

52 dmod%.o: $(GENUNIX_DIR)/%.c
53     $(COMPILE.c) -o $@ $<
54     $(CTFCONVERT_O)

56 dmod%.ln: $(GENUNIX_DIR)/%.c
57     $(LINT.c) -c $<
```

```
*****
```

```
3155 Sun Sep 6 19:01:39 2015
```

```
new/usr/src/cmd/sgs/lex/Makefile.com
```

```
6117 Many small bugs prevent a clean build on SPARC
```

```
*****
```

```
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 2015 Gary Mills
23 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #

27 PROG= lex

29 MACHOBJJS= main.o subl.o sub2.o sub3.o header.o parser.o
30 OBJECTS= $(MACHOBJJS)
31 POBJS= $(OBJECTS:%=objs/%)

33 LIBRARY= libl.a
34 VERS= .1

36 LIBOBJJS= allprint.o libmain.o reject.o yyless.o yywrap.o
37 LIBOBJJS_W= allprint_w.o reject_w.o yyless_w.o
38 LIBOBJJS_E= reject_e.o yyless_e.o
39 OBJECTS= $(LIBOBJJS) $(LIBOBJJS_W) $(LIBOBJJS_E)

41 FORMS= nceucform ncform nrform

43 include ../../../../lib/Makefile.lib

45 SRCDIR = ../common

47 C99MODE= $(C99_ENABLE)

49 CERRWARN += -gcc=-Wno-unused-label
50 CERRWARN += -gcc=-Wno-uninitialized
51 CERRWARN += -gcc=-Wno-parentheses

53 # Override default source file derivation rule (in Makefile.lib)
54 # from objects
55 #
56 MACHSRCS= $(MACHOBJJS:%.o=../common/%.c)
57 LIBSRCS = $(LIBOBJJS:%.o=../common/%.c)
58 SRCS= $(MACHSRCS) $(LIBSRCS)

60 LIBS = $(DYNLIB) $(LINTLIB)
```

```
62 # Append to LINTFLAGS and LINTFLAGS64 from lib/Makefile.lib
63 LINTFLAGS += -erroff=E_NAME_MULTIPLY_DEF2 -erroff=E_FUNC_RET_MAYBE_IGNORED2
64 LINTFLAGS64 += -erroff=E_NAME_MULTIPLY_DEF2 -erroff=E_FUNC_RET_MAYBE_IGNORED2

66 LINTSRCS= ../common/libl-1$(LIBNAME)
67 $(LINTLIB):= SRCS = $(SRCDIR)/$(LINTSRC)

69 INCLIST= $(INCLIST_$(MACH)) -I../include -I../include/$(MACH)
70 DEFLIST= -DELF

72 # It is not very clean to base the conditional definitions as below, but
73 # this will have to do for now.
74 #
75 #$(LIBOBJJS_W):= DEFLIST = -DEUC -DJLSLEX -DWOPTION -D$*=$*_w
76 objs/%_w.o:= DEFLIST = -DEUC -DJLSLEX -DWOPTION -D$*=$*_w
77 pics/%_w.o:= DEFLIST = -DEUC -DJLSLEX -DWOPTION -D$*=$*_w

79 #$(LIBOBJJS_E):= DEFLIST = -DEUC -DJLSLEX -DEOPTION -D$*=$*_e
80 objs/%_e.o:= DEFLIST = -DEUC -DJLSLEX -DEOPTION -D$*=$*_e
81 pics/%_e.o:= DEFLIST = -DEUC -DJLSLEX -DEOPTION -D$*=$*_e

83 CPPFLAGS= $(INCLIST) $(DEFLIST) $(CPPFLAGS.master)
84 BUILD.AR= $(AR) $(ARFLAGS) @$@ '$(ORDER)' $(OBJ) | '$(TSORT)\'

79 LINTFLAGS= -amux
86 LINTPOUT= lint.out

82 $(LINTLIB):= LINTFLAGS = -nvx
88 $(ROOTPROG):= FILEMODE = 0555

90 ROOTFORMS= $(FORMS:%=$(ROOTSHLIBCCS)/%)

92 ROOTLINTDIR= $(ROOTLIBDIR)
93 ROOTLINT= $(LINTSRCS:../common/%=$(ROOTLINTDIR)/%)

95 DYNLINKLIBDIR= $(ROOTLIBDIR)
96 DYNLINKLIB= $(LIBLINKS:%=$(DYNLINKLIBDIR)/%)

98 # Need to make sure lib-make's are warning free
99 $(DYNLIB) := CFLAGS += $(CCVERBOSE)
100 $(DYNLIB) := CFLAGS64 += $(CCVERBOSE)

102 LDLIBS += -lc
97 $(DYNLIB) := LDLIBS += -lc

104 CLEANFILES += ../common/parser.c $(LINTPOUT)
105 CLOBBERFILES += $(LIBS) $(LIBRARY)

107 lint: lintcheck
```

```

*****
2594 Sun Sep 6 19:01:39 2015
new/usr/src/cmd/sgs/lex/Makefile.targ
6117 Many small bugs prevent a clean build on SPARC
*****
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 2015 Gary Mills
23 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
26 # ident "%Z%M% %I% %E% SMI"

28 $(PROG): objs .WAIT $(POBJS)
29 $(LINK.c) $(POBJS) -o $@ $(LDLIBS)
30 $(POST_PROCESS)

32 # conditional assignment of default permissions for the installed
33 # DYNLIB/DYNLIBCCC. Set here and in Makefile.mach, instead of Makefile.lib,
34 # because ROOTLIBDIR may be changed in the including makefile after it has
35 # included Makefile.lib.
36 #
37 $(ROOTLIBDIR)/$(DYNLIB) := FILEMODE= 755
38 $(ROOTLIBDIR)/$(DYNLIBCCC) := FILEMODE= 755

40 $(ROOTLIBDIR64)/$(DYNLIB) := FILEMODE= 755
41 $(ROOTLIBDIR64)/$(DYNLIBCCC) := FILEMODE= 755

43 # library install rules
44 $(ROOTLIBDIR)/%: %
45 $(INS.file)
46 $(ROOTLIBDIR64)/%: %
47 $(INS.file)

49 $(ROOTLIBDIR)/$(LIBLINKS): $(ROOTLIBDIR)/$(LIBLINKS)$(VERS)
50 $(INS.liblink)
51 $(ROOTLIBDIR64)/$(LIBLINKS): $(ROOTLIBDIR64)/$(LIBLINKS)$(VERS)
52 $(INS.liblink64)

54 $(ROOTLIBDIR)/$(LIBLINKSCCC): $(ROOTLIBDIR)/$(LIBLINKSCCC)$(VERS)
55 $(INS.liblinkccc)
56 $(ROOTLIBDIR64)/$(LIBLINKSCCC): $(ROOTLIBDIR64)/$(LIBLINKSCCC)$(VERS)
57 $(INS.liblinkccc64)

59 objs pics:
60 -@mkdir -p $@

```

```

62 $(LIBRARY): objs .WAIT $$ (OBJS)
63 $(BUILD.AR)
64 $(POST_PROCESS_A)

66 $(DYNLIB): pics .WAIT $$ (PICS)
67 $(BUILD.SO)
68 $(POST_PROCESS_SO)

70 $(DYNLIBCCC): pics .WAIT $$ (PICS)
71 $(BUILDCCC.SO)
72 $(POST_PROCESS_SO)

74 objs/%_w.o pics/%_w.o objs/%_e.o pics/%_e.o: ../common/%.c
75 $(COMPILE.c) -o $@ $<
76 $(POST_PROCESS_O)

78 objs/%.o pics/%.o: ../common/%.c
79 $(COMPILE.c) -o $@ $<
80 $(POST_PROCESS_O)

82 # install rule for lint library target
83 $(ROOTLINTDIR)/%: ../common/%
84 $(INS.file)

86 $(DYNLINKLIBDIR)/%: %$(VERS)
87 $(INS.liblink)

89 clean:
90 $(RM) $(POBJS) $(OBJS) $(PICS) $(CLEANFILES)

92 $(LINTLIB): $$ (SRCS)
93 $(LINT.c) -o $(LIBNAME) $(SRCS) > $(LINTOUT) 2>&1
92 lint: $(LINTPOUT) $(LINTLIB)

95 lintcheck: $$ (SRCS)
96 $(LINT.c) $(LINTCHECKFLAGS) $(SRCS) $(LDLIBS)
94 LINT_HDR= perl ../../tools/lint_hdr.pl

96 $(LINTPOUT): $(SRCS)
97 $(LINT_HDR) $(PROG) > $(LINTPOUT)
98 $(LINT_HDR) -s lex >> $(LINTPOUT)
99 $(LINT.c) $(MACHSRCS) $(LDLIBS) 2>&1 | tee -a $(LINTPOUT)

101 $(LINT_HDR) -s libl.so.1 >> $(LINTPOUT)
102 $(LINT.c) $(LIBSRCS) $(LDLIBS) 2>&1 | tee -a $(LINTPOUT)

104 $(LINTLIB): $(LINTSRCS)
105 $(LINT.c) -o $(LIBNAME) $(LINTSRCS)

```

```

*****
3295 Sun Sep 6 19:01:39 2015
new/usr/src/cmd/sgs/libelf/Makefile.com
6117 Many small bugs prevent a clean build on SPARC
*****
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 2015 Gary Mills
23 # Copyright (c) 1990, 2010, Oracle and/or its affiliates. All rights reserved.
24 #

26 LIBRARY=      libelf.a
27 VERS=         .1
28 M4=           m4

29 MACHOBJS=
30 COMOBJS=      ar.o      begin.o      cntl.o      cook.o \
31              data.o      end.o        fill.o      flag.o \
32              getarhdr.o  getarsym.o  getbase.o  getdata.o \
33              getehdr.o  getident.o  getphdr.o  getscn.o \
34              getshdr.o \
35              getphnum.o  getshnum.o  getshstrndx.o \
36              hash.o      input.o     kind.o \
37              ndxscn.o   newdata.o  newehdr.o  newphdr.o \
38              newscn.o   next.o     nextscn.o  output.o \
39              rand.o     rawdata.o  rawput.o  rawput.o \
40              strprtr.o  update.o   error.o    gelf.o \
41              clscook.o  checksum.o newehdr64.o newphdr64.o update64.o \
42 CLASSOBJS=
43              checksum64.o
44 BLTOBJS=      msg.o        xlate.o     xlate64.o
45 MISC OBJJS=   String.o    args.o      demangle.o  nlist.o \
46              nplist.o
47 MISC OBJJS64= nlist.o

49 OBJECTS=     $(BLTOBJS) $(MACHOBJS) $(COMOBJS) $(CLASSOBJS) $(MISC OBJJS)

51 include $(SRC)/lib/Makefile.lib

53 # Use the value of M4 set in Makefile.master via Makefile.lib

55 DEMOFILES=   Makefile      00README    acom.c      dcom.c \
56              pcom.c      tpcom.c     dispsyms.c
57 DEMOFILESRCDIR= ./demo
58 ROOTDEMODIRBASE=$(ROOT)/usr/demo/ELF
59 ROOTDEMODIRS= $(ROOTDEMODIRBASE)

```

```

61 include $(SRC)/cmd/sgs/Makefile.com

63 WARLOCKFILES= $(OBJECTS:%.o=wlocks/%.ll)

65 MAPFILES =     ../common/mapfile-vers

67 DYNFLAGS +=    $(VERSREF)
68 LDLIBS +=      $(CONVLIBDIR) $(CONV_LIB) -lc

70 LINTFLAGS +=   -u -erroff=E_BAD_PTR_CAST_ALIGN
71 LINTFLAGS64 += -u -erroff=E_CAST_INT_TO_SMALL_INT

73 CERRWARN +=    -_gcc=-Wno-parentheses
74 CERRWARN +=    -_gcc=-Wno-uninitialized

76 BUILD.AR=     $(RM) $@ ; \
77               $(AR) q $@ '$(LORDER) $(OBJECTS:%=$(DIR)/%)' $(TSORT) \
78               $(POST_PROCESS_A)

81 BLTDEFS=       msg.h
82 BLTDATA=       msg.c
83 BLTMESG=       $(SGSMSGDIR)/libelf

85 BLTFILES=     $(BLTDEFS) $(BLTDATA) $(BLTMESG)

87 SGSMSGCOM=    ../common/libelf.msg
88 SGSMSG32=     ../common/libelf.32.msg
89 SGSMSGTARG=   $(SGSMSGCOM)
90 SGSMSGALL=    $(SGSMSGCOM) $(SGSMSG32)

92 SGSMSGFLAGS1= $(SGSMSGFLAGS) -m $(BLTMESG)
93 SGSMSGFLAGS2= $(SGSMSGFLAGS) -h $(BLTDEFS) -d $(BLTDATA) -n libelf_msg

95 BLTSRCS=      $(BLTOBJS:%.o=%.c)
96 LIBSRCS=      $(COMOBJS:%.o=../common/%.c) $(MISC OBJJS:%.o=../misc/%.c) \
97               $(MACHOBJS:%.o=%.c) $(BLTSRCS)
98 SRCS=         ../common/libelf
99 LINTSRCS=     $(LIBSRCS) ../common/lintsup.c

101 ROOTFS_DYNLIB= $(DYNLIB:%=$(ROOTFS_LIBDIR)/%)
102 ROOTFS_LINTLIB= $(LINTLIB:%=$(ROOTFS_LIBDIR)/%)

104 ROOTFS_DYNLIB64= $(DYNLIB:%=$(ROOTFS_LIBDIR64)/%)
105 ROOTFS_LINTLIB64= $(LINTLIB:%=$(ROOTFS_LIBDIR64)/%)

107 $(ROOTFS_DYNLIB) := FILEMODE= 755
108 $(ROOTFS_DYNLIB64) := FILEMODE= 755

110 LIBS =        $(DYNLIB) $(LINTLIB)

112 CLEANFILES += $(LINTOUTS) $(BLTSRCS) $(BLTFILES) $(WARLOCKFILES)

114 .PARALLEL:   $(LIBS)

```

```

*****
2500 Sun Sep 6 19:01:40 2015
new/usr/src/cmd/sgs/yacc/Makefile.com
6117 Many small bugs prevent a clean build on SPARC
*****
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 2015 Gary Mills
23 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #

27 PROG=          yacc

29 COMOBJS=       y1.o y2.o y3.o y4.o
30 OBJECTS=       $(COMOBJS)
31 POBJS=         $(OBJECTS:%=objs%)

33 OBJECTS=       libmai.o libzer.o

35 LIBRARY=       liby.a
36 VERS=         .1
37 YACCPAR=       yaccpar

39 include ../../../../lib/Makefile.lib

41 SRCDIR =        ../common

43 # Override default source file derivation rule (in Makefile.lib)
44 # from objects
45 #
46 COMSRCS=       $(COMOBJS:%.o=../common/%.c)
47 LIBSRCS=       $(OBJECTS:%.o=../common/%.c)
48 SRCS=          $(COMSRCS) $(LIBSRCS)

50 LIBS =         $(DYNLIB) $(LINTLIB)

52 # Append to LINTFLAGS and LINTFLAGS64 from lib/Makefile.lib
53 LINTFLAGS +=   -erroff=E_NAME_MULTIPLY_DEF2
54 LINTFLAGS64 += -erroff=E_NAME_MULTIPLY_DEF2

56 # Tune ZDEFS to ignore undefined symbols for building the yacc shared library
57 # since these symbols (mainly yyparse) are to be resolved elsewhere.
58 #
59 $(DYNLIB):= ZDEFS = $(ZNODEFS)
60 $(DYNLIBBCC):= ZDEFS = $(ZNODEFS)
61 LINTSRCS=      ../../common/lib-1$(LIBNAME)

```

```

62 $(LINTLIB):=   SRCS = $(SRCDIR)/$(LINTSRC)

64 INCLIST=       -I../include -I../include/$(MACH)
65 CPPFLAGS=      $(INCLIST) $(DEFLIST) $(CPPFLAGS.master)
66 $(PROG):=      LDLIBS = $(LDLIBS.cmd)
67 LDLIBS=        $(LDLIBS.cmd)
68 BUILD.AR=      $(AR) $(ARFLAGS) $@ `$(LORDER) $(OBJS) | $(TSORT)`

62 LINTFLAGS=     -amux
69 LINTPOUT=      lint.out

71 C99MODE=       $(C99_ENABLE)
72 CFLAGS +=      $(CCVERBOSE)
73 CFLAGS64 +=    $(CCVERBOSE)
74 CERRWARN +=    -_gcc=-Wno-parentheses
75 CERRWARN +=    -_gcc=-Wno-uninitialized

71 $(LINTLIB):=   LINTFLAGS = -nvx
77 $(ROOTPROG):= FILEMODE = 0555

79 ROOTYACCPAR=   $(YACCPAR:%=$(ROOTSHLIBCCS)/%)

81 ROOTLINTDIR=   $(ROOTLIBDIR)
82 ROOTLINT=      $(LINTSRCS:../common/%=$(ROOTLINTDIR)/%)

84 DYNLINKLIBDIR= $(ROOTLIBDIR)
85 DYNLINKLIB=    $(LIBLINKS:%=$(DYNLINKLIBDIR)/%)

87 LDLIBS +=     -lc
82 $(DYNLIB) :=   LDLIBS += -lc

89 CLEANFILES += $(LINTPOUT)
90 CLOBBERFILES += $(LIBS) $(LIBRARY)

92 lint: lintcheck

```

new/usr/src/cmd/sgs/yacc/Makefile.targ

1

```
*****
2493 Sun Sep 6 19:01:40 2015
new/usr/src/cmd/sgs/yacc/Makefile.targ
6117 Many small bugs prevent a clean build on SPARC
*****
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 2015 Gary Mills
23 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
25 # ident "%Z%M% %I% %E% SMI"
26 #

28 $(PROG): objs .WAIT $(POBJS)
29 $(LINK.c) $(POBJS) -o $@ $(LDLIBS)
30 $(POST_PROCESS)

32 # conditional assignment of default permissions for the installed
33 # DYNLIB/DYNLIBCCC. Set here and in Makefile.mach, instead of Makefile.lib,
34 # because ROOTLIBDIR may be changed in the including makefile after it has
35 # included Makefile.lib.
36 #
37 $(ROOTLIBDIR)/$(DYNLIB) := FILEMODE= 755
38 $(ROOTLIBDIR)/$(DYNLIBCCC) := FILEMODE= 755

40 $(ROOTLIBDIR64)/$(DYNLIB) := FILEMODE= 755
41 $(ROOTLIBDIR64)/$(DYNLIBCCC) := FILEMODE= 755

43 # library install rules
44 $(ROOTLIBDIR)/%: %
45 $(INS.file)
46 $(ROOTLIBDIR64)/%: %
47 $(INS.file)

49 $(ROOTLIBDIR)/$(LIBLINKS): $(ROOTLIBDIR)/$(LIBLINKS)$(VERS)
50 $(INS.liblink)
51 $(ROOTLIBDIR64)/$(LIBLINKS): $(ROOTLIBDIR64)/$(LIBLINKS)$(VERS)
52 $(INS.liblink64)

54 $(ROOTLIBDIR)/$(LIBLINKSCCC): $(ROOTLIBDIR)/$(LIBLINKSCCC)$(VERS)
55 $(INS.liblinkccc)
56 $(ROOTLIBDIR64)/$(LIBLINKSCCC): $(ROOTLIBDIR64)/$(LIBLINKSCCC)$(VERS)
57 $(INS.liblinkccc64)

59 objs pics:
60 -@mkdir -p $@
```

new/usr/src/cmd/sgs/yacc/Makefile.targ

2

```
62 $(LIBRARY): objs .WAIT $$ (OBJS)
63 $(BUILD.AR)
64 $(POST_PROCESS_A)

66 $(DYNLIB): pics .WAIT $$ (PICS)
67 $(BUILD.SO)
68 $(POST_PROCESS_SO)

70 $(DYNLIBCCC): pics .WAIT $$ (PICS)
71 $(BUILDCCC.SO)
72 $(POST_PROCESS_SO)

74 objs/%.o pics/%.o: ../common/%.c
75 $(COMPILE.c) -o $@ $<
76 $(POST_PROCESS_O)

78 # install rule for lint library target
79 $(ROOTLINTDIR)/%: ../common/%
80 $(INS.file)

82 $(DYNLINKLIBDIR)/%: %$(VERS)
83 $(INS.liblink)

85 clean:
86 $(RM) $(POBJS) $(OBJS) $(PICS) $(CLEANFILES)

88 $(LINTLIB): $$ (SRCS)
89 $(LINT.c) -o $(LIBNAME) $(SRCS) > $(LINTOUT) 2>&1
88 lint: $(LINTPOUT) $(LINTLIB)

91 lintcheck: $$ (SRCS)
92 $(LINT.c) $(LINTCHECKFLAGS) $(SRCS) $(LDLIBS)

91 LINT_HDR= perl ../../tools/lint_hdr.pl

93 $(LINTPOUT): $(SRCS)
94 $(LINT_HDR) $(PROG) > $(LINTPOUT)
95 $(LINT_HDR) -s yacc >> $(LINTPOUT)
96 $(LINT.c) $(COMSRCS) $(LDLIBS) 2>&1 | tee -a $(LINTPOUT)

98 $(LINT_HDR) -s liby.so.1 >> $(LINTPOUT)
99 $(LINT.c) $(LIBSRCS) $(LDLIBS) 2>&1 | tee -a $(LINTPOUT)

102 $(LINTLIB): $(LINTSRCS)
103 $(LINT.c) -o $(LIBNAME) $(LINTSRCS)
```



```

*****
14029 Sun Sep 6 19:01:40 2015
new/usr/src/lib/Makefile
6117 Many small bugs prevent a clean build on SPARC
*****
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 2011 Nexenta Systems, Inc. All rights reserved.
23 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright (c) 2012 by Delphix. All rights reserved.
25 # Copyright (c) 2012, Joyent, Inc. All rights reserved.
26 # Copyright (c) 2013 Gary Mills
27 # Copyright 2014 Garrett D'Amore <garrett@damore.org>
28 # Copyright (c) 2015 Gary Mills
29 #
30 include ../Makefile.master
31 #
32 # Note that libcurses installs commands along with its library.
33 # This is a minor bug which probably should be fixed.
34 # Note also that a few extra libraries are kept in cmd source.
35 #
36 # Certain libraries are linked with, hence depend on, other libraries.
37 #
38 # Although we have historically used .WAIT to express dependencies, it
39 # reduces the amount of parallelism and thus lengthens the time it
40 # takes to build the libraries. Thus, we now require that any new
41 # libraries explicitly call out their dependencies. Eventually, all
42 # the library dependencies will be called out explicitly. See
43 # "Library interdependencies" near the end of this file.
44 #
45 # Aside from explicit dependencies (and legacy .WAITs), all libraries
46 # are built in parallel.
47 #
48 .PARALLEL:
49 #
50 SUBDIRS= \
51     common .WAIT \
52     ../cmd/sgs/libconv \
53     ../cmd/sgs/libdl .WAIT
54 #
55 SUBDIRS += \
56     libc .WAIT \
57     ../cmd/sgs/libelf .WAIT \
58     c_synonyms \
59     libmd \
60     libmd5 \
61     librsm

```

```

62     libmp .WAIT \
63     libnsl \
64     libsecdb .WAIT \
65     librpcsvc \
66     libsocket .WAIT \
67     libsctp \
68     libsip \
69     libcommutil \
70     libresolv \
71     libresolv2 .WAIT \
72     libw .WAIT \
73     libintl .WAIT \
74     ../cmd/sgs/librtld_db \
75     libaio \
76     libast \
77     libdll \
78     libcmd \
79     libshell \
80     libsum \
81     librt \
82     libadm \
83     libctf \
84     libdtrace \
85     libdtrace_jni \
86     libcurses \
87     libtermcap \
88     libgen \
89     libgss \
90     libpam \
91     libuuid \
92     libthread \
93     libpthread .WAIT \
94     libslp \
95     libbsdmalloc \
96     libdoor \
97     libdevinfo \
98     libldadm \
99     libdlpi \
100    libeti \
101    libcrypt \
102    libdns_sd \
103    libefi \
104    libfstyp \
105    libwanboot \
106    libwanbootutil \
107    libcryptoutil \
108    libinetutil \
109    libipadm \
110    libipd \
111    libipmp \
112    libiscsit \
113    libkmf \
114    libkstat \
115    libkvm \
116    liblm \
117    libmalloc \
118    libmapmalloc \
119    libmtmalloc \
120    libnls \
121    libnwam \
122    libsbmbios \
123    libtecla \
124    libumem \
125    libnvpair .WAIT \
126    libexacct \
127    libsas1 \

```

new/usr/src/lib/Makefile

```

128     libldap5           \
129     libldap            .WAIT \
130     libbsm             \
131     libsys             \
132     libsysevent       \
133     libnisdb           \
134     libpool            \
135     libpp              \
136     libproc           \
137     libproject        \
138     libsendfile       \
139     nametoaddr        \
140     ncad_addr         \
141     hbaapi            \
142     smhba             \
143     sun_fc            \
144     sun_sas           \
145     gss_mechs/mech_krb5 .WAIT \
146     libkrb5           .WAIT \
147     krb5              .WAIT \
148     libsmbfs          \
149     libfcoe           \
150     libsrpt           \
151     libstmf           \
152     libstmfproxy     \
153     libnsctl         \
154     libunistat       \
155     libdscfg         \
156     librbc            \
157     libinstzones     \
158     libpkg            \
159     libpcidb         \
160     libml             \
161     libm              \
162     libmvec          \
\
165 SUBDIRS += \
166     passwdutil       \
167     pam_modules      \
168     crypt_modules    \
169     libadt_jni       \
170     abi              \
171     auditd_plugins  \
172     libvolmgt        \
173     libdevice        \
174     libdevid         \
175     libc_db          \
176     libndmp          \
177     libsec           \
178     libtnfprobe     \
179     libtnf           \
180     libtnfctl        \
181     libdhcpageant   \
182     libdhcputil     \
183     libxnet          \
184     libipsecutil    \
185     nsswitch         \
186     print            \
187     libuutil         \
188     libscf           \
189     libinetsvc       \
190     librestart       \
191     libsched         \
192     libelfsign       \
193     pkcs11          .WAIT \

```

3

new/usr/src/lib/Makefile

```

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

```

4

new/usr/src/lib/Makefile

```

260     scsi          \
261     libima        \
262     libsun_ima    \
263     mpapi         \
264     librstp      \
265     libreparse    \
266     libhotplug    \
267     libfruutils   .WAIT  \
268     libfru        \
269     ${$(MACH)_SUBDIRS}

271 i386_SUBDIRS=    \
272     libfdisk      \
273     libsaveargs

275 sparc_SUBDIRS= .WAIT \
276     efcode        \
277     libds         \
278     libdscp       \
279     libprtdiag    .WAIT  \
280     libprtdiag_psr \
281     libpri        \
282     librsc        \
283     storage       \
284     libpcp        \
285     libtsalarm    \
286     libvl2n

288 FM_sparc_DEPLIBS= libpri

290 fm:
291     libexacct     \
292     libipmi       \
293     libzfs        \
294     scsi          \
295     ${FM_${$(MACH)_DEPLIBS}

297 #
298 # Create a special version of $(SUBDIRS) with no .WAIT's, for use with the
299 # clean and clobber targets (for more information, see those targets, below).
300 #
301 NOWAIT_SUBDIRS= $(SUBDIRS:.WAIT=)

303 DCSSUBDIRS =    \
304     lvm

306 MSGSUBDIRS=    \
307     abi           \
308     auditd_plugins \
309     brand         \
310     cfgadm_plugins \
311     gss_mechs/mech_dh \
312     gss_mechs/mech_krb5 \
313     krb5          \
314     libast        \
315     libbsm        \
316     libc          \
317     libcfgadm     \
318     libcmd        \
319     libcontract   \
320     libcurses     \
321     libdhcputil   \
322     libipseutil   \
323     libdiskmgt    \
324     libdladm      \
325     libdll        \

```

5

new/usr/src/lib/Makefile

```

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

371 sparc_MSGSUBDIRS= \
372     libprtdiag    \
373     libprtdiag_psr

375 i386_MSGSUBDIRS= libfdisk

377 HDRSUBDIRS=    \
378     auditd_plugins \
379     libast         \
380     libbrand       \
381     libbsm        \
382     libc           \
383     libcmd         \
384     libcmdutils   \
385     libcommputil  \
386     libcontract   \
387     libcpc        \
388     libctf        \
389     libcurses     \
390     libtermcap    \
391     libcryptoutil \

```

6

new/usr/src/lib/Makefile

```

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

```

7

new/usr/src/lib/Makefile

```

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

509 i386_HDRSUBDIRS= //
510 libfdisk //
511 libsaveargs //

513 sparc_HDRSUBDIRS= //
514 libds //
515 libdscp //
516 libpri //
517 libvl2n //
518 storage //

520 all := TARGET= all
521 check := TARGET= check
522 clean := TARGET= clean
523 clobber := TARGET= clobber

```

8

```

524 install := TARGET= install
525 install_h := TARGET= install_h
526 lint := TARGET= lint
527 _dc := TARGET= _dc
528 _msg := TARGET= _msg

530 .KEEP_STATE:

532 #
533 # For the all and install targets, we clearly must respect library
534 # dependencies so that the libraries link correctly. However, for
535 # the remaining targets (check, clean, clobber, install_h, lint, _dc
536 # and _msg), libraries do not have any dependencies on one another
537 # and thus respecting dependencies just slows down the build.
538 # As such, for these rules, we use pattern replacement to explicitly
539 # avoid triggering the dependency information. Note that for clean,
540 # clobber and lint, we must use $(NOWAIT_SUBDIRS) rather than
541 # $(SUBDIRS), to prevent '.WAIT' from expanding to '.WAIT-noddepend'.
542 #

544 all: $(SUBDIRS)

546 install: $(SUBDIRS) .WAIT install_extra

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

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

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

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

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

561 #
562 # Library interdependencies are called out explicitly here
563 #
564 auditd_plugins: libbsm libnsl libsecdb
565 krb5: gss_mechs/mech_krb5
566 gss_mechs/mech_krb5: libgss libnsl libsocket libresolv pkcs11
567 gss_mechs/mech_spnego: gss_mechs/mech_krb5
568 libadt_jni: libbsm
569 libast: libsocket libm
570 libadutils: libldap5 libresolv libsocket libnsl
571 nsswitch: libadutils libidmap
572 libbe: libzfs
573 libbsm: libtsol
574 libcmd: libsum libast libsocket libnsl
575 libcmdutils: libavl libnvpair
576 libcmdutils: libavl
577 libcontract: libnvpair
578 libpcp: libdevinfo
579 libdevinfo: libdevinfo
580 libdevinfo: libnvpair libsec
581 libdhcagent: libsocket libdhcputil libuuid libdipi libcontract
582 libdhcputil: libnsl libgen libinetutil libdipi
583 libdladm: libdevinfo libinetutil libsocket libscf librcm libnvpair \
584     libxacct libnsl libkstat libcurses
584 libdevice: libnvpair
585 libdll: libast
586 libdipi: libinetutil libdladm
587 libds: libsysevent
588 libdscfg: libnsctl libunistat libsocket libnsl

```

```

589 libdtrace: libproc libgen libctf libmapmalloc
584 libdtrace: libproc libgen libctf
590 libdtrace_jni: libuutil libdtrace
591 libefi: libuuid
592 libfcoe: libdladm
593 libfstyp: libnvpair
594 libelfsign: libcryptoutil libkmf
595 libidmap: libadutils libldap5 libavl libsldap libuutil
596 libipadm: libnsl libinetutil libsocket libdipi libnvpair libdhcagent \
597     libdladm libsecdb
598 libiscsit: libc libnvpair libstmf libuuid libnsl
599 libkfm: libcryptoutil pkcs11
600 libm: libc
601 libml: libc libm
602 libmvec: libc libm
603 libnsl: libmd5
604 libmapid: libresolv
605 libnisdb: libnsl libldap5
606 librcm: libnvpair
607 librdc: libsocket libnsl libnsctl libunistat libdscfg
608 libuutil: libdipi
609 libinetutil: libsocket
610 libipseutil: libtecla libsocket
611 libinstzones: libzonecfg libcontract
612 libpkg: libwanboot libscf libadm
613 libnwm: libscf libbsm libdladm
605 libnwm: libscf
614 libsecdb: libnsl
615 libsas1: libgss libsocket pkcs11 libmd
616 sasl_plugins: pkcs11 libgss libsocket libsas1
617 libscfp: libsocket
618 libshell: libast libcmd libdll libsocket libsecdb libm
619 libsip: libmd5
620 libsbmfs: libcmdutils libsocket libnsl libkrb5 libsec libidmap
621 libsbmfs: libcmdutils libsocket libnsl libkrb5
621 libsocket: libnsl
622 libstmfproxy: libstmf libsocket libnsl libpthread
623 libsum: libast
624 libsysevent: libsecdb libnvpair
616 libsysevent: libsecdb
625 libldap5: libsas1 libsocket libnsl libmd
626 libldap: libldap5 libtsol libnsl libc libscf libresolv
627 libpool: libnvpair libxacct
628 libpp: libast
629 libzonecfg: libc libsocket libnsl libuuid libnvpair libsysevent libsec \
630     libbrand libpool libscf
631 libproc: ../cmd/sgs/librtld_db ../cmd/sgs/libelf libctf libsavargs
632 libproject: libpool libproc libsecdb
633 libtermcap: libcurses
634 libtsnet: libnsl libtsol libsecdb
635 libwrap: libnsl libsocket
636 libwanboot: libnvpair libresolv libnsl libsocket libdevinfo libinetutil \
637     libdhcputil
638 libwanbootutil: libnsl
639 pam_modules: libproject passwdutil smbsrv libtsnet
640 libscf: libuutil libmd libgen libsbmfs libnsl libnvpair
631 pam_modules: libproject passwdutil smbsrv
632 libscf: libuutil libmd libgen libsbmfs libnsl
641 libnetsvc: libscf
642 librestart: libuutil libscf libpool libproject libsecdb libnvpair
634 librestart: libuutil libscf
643 libsavargs: libdisasm
644 ../cmd/sgs/libdl: ../cmd/sgs/libconv
645 ../cmd/sgs/libelf: ../cmd/sgs/libconv
646 passwdutil: libsldap
647 pkcs11: libcryptoutil libmd

```

```

648 # Adding libuuid or libldadm results in a circular dependency
648 pkcs11: libcryptoutil
649 print: libldap5
650 udapl/udapl_tavor: udapl/libdat
651 libzfs: libdevid libgen libnvpair libuutil \
652 libadm libavl libefi libidmap libmd libzfs_core libm
653 libzfs_core: libnvpair
654 libzfs_jni: libdiskmgt libnvpair libzfs
655 libzpool: libavl libumem libnvpair libcmdutils
656 libsec: libavl libidmap
657 brand: libc libsocket
658 libshare: libscf libzfs libuuid libfsmgt libsecdb libumem libsmbf
659 libxacct/demo: libxacct libproject libsocket libnsl
660 libtsalarm: libpcp
661 smbsrv: libsocket libnsl libmd libxnet libpthread librt \
662 libshare libidmap pkcs11 libsqlite libcryptoutil \
663 librepase libcmdutils libfakekernel
664 libv12n: libds libuuid
665 libvrrpadm: libsocket libldadm libscf
666 libvscan: libscf
667 libfru: libfruutils
668 scsi: libnvpair libfru
669 mpapi: libpthread libdevinfo libsysevent libnvpair
670 sun_fc: libdevinfo libsysevent libnvpair
671 libsun_ima: libdevinfo libsysevent libnsl
672 sun_sas: libdevinfo libsysevent libnvpair libkstat libdevid
673 libgrubmgt: libdevinfo libzfs libfstyp
674 pylibbe: libbe libzfs
675 pyzfs: libnvpair libzfs
676 pysolaris: libsec libidmap
677 libreparse: libnvpair
678 libhotplug: libnvpair
679 cfgadm_plugins: libhotplug
680 libilb: libsocket
681 libipmi: libm
682 libprtdiag: libm
683 libsqlite: libm
684 libstmf: libm
685 libvscan: libm

```

```
688 $(INTEL_BUILD)libdiskmgt:libfdisk
```

```

690 #
691 # The reason this rule checks for the existence of the
692 # Makefile is that some of the directories do not exist
693 # in certain situations (e.g., exportable source builds,
694 # OpenSolaris).
695 #

```

```

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

```

```

703 $(SUBDIRS:%=%-nodepend):
704     @if [ -f $@:%-nodepend=%/Makefile ]; then \
705         cd $@:%-nodepend=%; pwd; $(MAKE) $(TARGET); \
706     else \
707         true; \
708     fi

```

```
710 FRC:
```

new/usr/src/lib/libcurses/screen/keyname.sh

1

```
*****
2541 Sun Sep 6 19:01:40 2015
new/usr/src/lib/libcurses/screen/keyname.sh
6117 Many small bugs prevent a clean build on SPARC
*****
1 #
2 # CDDL HEADER START
3 #
4 # The contents of this file are subject to the terms of the
5 # Common Development and Distribution License, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 #
23 # Copyright 2015 Gary Mills
24 # Copyright 1997 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #
27 #
28 #
29 # Copyright (c) 1988 AT&T
30 # All Rights Reserved
31 #
32 #
33 #
34 # University Copyright- Copyright (c) 1982, 1986, 1988
35 # The Regents of the University of California
36 # All Rights Reserved
37 #
38 # University Acknowledgm- Portions of this document are derived from
39 # software developed by the University of California, Berkeley, and its
40 # contributors.
41 #
42 #
43 #ident "%Z%M% %I% %E% SMI"
44 rm -f keyname.c
45 /usr/bin/print "#include \"curses_inc.h\" \n" > keyname.c
46 /usr/bin/print "static char *keystrings[] = \n\t\t{ \" >> keyname.c
47 /usr/bin/echo "#include \"curses_inc.h\" \n" > keyname.c
48 /usr/bin/echo "static char *keystrings[] = \n\t\t{ \" >> keyname.c
49 {
50 { grep -v 'KEY_F(' keycaps | awk '{ print $5, $4 }' | sed -e 's//g' -e 's/KE
51 # These three aren't in keycaps
52 echo '0401 BREAK\n0530 SRESET\n0531 RESET'
53 } | sort -n | awk '
unchanged_portion_omitted
```

```

*****
12762 Sun Sep 6 19:01:40 2015
new/usr/src/lib/libcurses/screen/maketerm.ed
6117 Many small bugs prevent a clean build on SPARC
*****
_____unchanged_portion_omitted_____

286 struct str4 {
287     charptr
288     .
289     l,$s/_Sentinel/Sentinel/
290     w ./tmp/term.h.new
291     e ./tmp/term.h.new
292     g/[      ]_cursor/_s/_cursor/_/crsr_/
293     g/[      ]_delete/_s/_delete/_/dlt_/
294     g/[      ]_enter/_s/_enter/_/entr_/
295     g/[      ]_insert/_s/_insert/_/ins_/
296     g/[      ]_key/_s/_key/_/ky_/
297     g/[      ]_keypad/_s/_keypad/_/kpad_/
298     g/[      ]_label/_s/_label/_/labl_/
299     g/[      ]_memory/_s/_memory/_/mem_/
300     g/[      ]_parm/_s/_parm/_/prm_/
301     g/[      ]_scroll/_s/_scroll/_/scrll_/
302     g/^      _.....[,;]          \\/s/, //
303     g/^      _.....[,;]          \\/s/, //
304     g/^      _.....[,;]          \\/s/, //
305     w ./tmp/term.h.new
306     e ./tmp/term.h.new
307     !# the following lines GO away when Vr2 compat code goes away
308     l;/^#define auto_left_margin/,/^#define xon_xoff/s/CURB.*/CUR_b/
309     l;/^#define auto_left_margin/,/^#define xon_xoff/w ./tmp/tmp.term.h
310     l;/^#define auto_left_margin/,/^#define xon_xoff/d
311     !/usr/bin/print '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
311     !/usr/bin/echo '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
312     .-lr !awk -f ./tmp/tmp.awk < ./tmp/tmp.term.h
313     l;/^#define columns/,/^#define width_status_line/s/CURN.*/CUR_c/
314     l;/^#define columns/,/^#define width_status_line/w ./tmp/tmp.term.h
315     l;/^#define columns/,/^#define width_status_line/d
316     !/usr/bin/print '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
316     !/usr/bin/echo '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
317     .-lr !awk -f ./tmp/tmp.awk < ./tmp/tmp.term.h
318     l;/^#define back_tab/,/^#define lab_f8/s/CURS.*/CUR_Vr2_Astrs._s/
319     l;/^#define back_tab/,/^#define lab_f8/w ./tmp/tmp.term.h
320     l;/^#define back_tab/,/^#define lab_f8/d
321     !/usr/bin/print '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
321     !/usr/bin/echo '{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
322     .-lr !awk -f ./tmp/tmp.awk < ./tmp/tmp.term.h
323     l;/^#define lab_f9/,/^#define prtr_non/s/CURS.*/CUR_Vr2_Bstrs._s/
324     l;/^#define lab_f9/,/^#define prtr_non/w ./tmp/tmp.term.h
325     l;/^#define lab_f9/,/^#define prtr_non/d
326     !/usr/bin/print 'BEGIN{i=100}\n{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
326     !/usr/bin/echo 'BEGIN{i=100}\n{printf "%s%d\n", $0,++i}' > ./tmp/tmp.awk
327     .-lr !awk -f ./tmp/tmp.awk < ./tmp/tmp.term.h
328     !rm -f ./tmp/tmp.term.h ./tmp/tmp.awk
329     w ./tmp/term.h.new
330     e ./tmp/term.h.new
331     !# Vr2 compat code ends here
332     li
333     #ifndef _TERM_H
334     #define _TERM_H

336 /*
337 * term.h - this file is automatically made from caps and maketerm.ed.
338 * Don't make changes directly to term.h.
339 */

```

```

341 #ifdef __cplusplus
342 extern "C" {
343 #endif

345 .
346 $a

348 struct _str_struct {
349     struct str4 str1;
350     struct str4 str2;
351     struct str4 str3;
352     struct str4 str4;
353 };
_____unchanged_portion_omitted_____

```