```
**********************************************************
   43894 Thu Jan 17 01:17:59 2019
new/usr/src/tools/cw/cw.c
jlevon comments
**********************************************************
______unchanged_portion_omitted_
```

```
597 static boolean_t
598 is_source_file(const char *path)
599 {
600         char *ext = strrchr(path, '.');

602         if ((ext == NULL) || ((ext + 1) == '\0'))
603                 return (B_FALSE);

605         ext += 1;

607         if ((strcasecmp(ext, "c") == 0) ||
608             (strcmp(ext, "cc") == 0) ||
609             (strcmp(ext, "i") == 0) ||
610             (strcasecmp(ext, "s") == 0) ||
611             (strcmp(ext, "cpp") == 0)) {
612                 return (B_TRUE);
613         }

615         return (B_FALSE);
616 }


619 #endif /* ! codereview */
620 static void
621 do_gcc(cw_ictx_t *ctx)
622 {
623         int c;
624         int nolibc = 0;
625         int in_output = 0, seen_o = 0, c_files = 0;
626         cw_op_t op = CW_O_LINK;
627         char *model = NULL;
628         char *nameflag;
629         int mflag = 0;

631         if (ctx->i_flags & CW_F_PROG) {
632                 newae(ctx->i_ae, "--version");
633                 return;
634         }

636         newae(ctx->i_ae, "-fident");
637         newae(ctx->i_ae, "-finline");
638         newae(ctx->i_ae, "-fno-inline-functions");
639         newae(ctx->i_ae, "-fno-builtin");
640         newae(ctx->i_ae, "-fno-asm");
641         newae(ctx->i_ae, "-fdiagnostics-show-option");
642         newae(ctx->i_ae, "-nodefaultlibs");

644 #if defined(__sparc)
645         /*
646          * The SPARC ldd and std instructions require 8-byte alignment of
647          * their address operand.  gcc correctly uses them only when the
648          * ABI requires 8-byte alignment; unfortunately we have a number of
649          * pieces of buggy code that doesn't conform to the ABI.  This
650          * flag makes gcc work more like Studio with -xmemalign=4.
651          */
652         newae(ctx->i_ae, "-mno-integer-ldd-std");
653 #endif

655         /*
```



```
656          * This is needed because 'u' is defined
657          * under a conditional on 'sun'.  Should
658          * probably just remove the conditional,
659          * or make it be dependent on '__sun'.
660          *
661          * -Dunix is also missing in enhanced ANSI mode
662          */
663         newae(ctx->i_ae, "-D__sun");

665         if (asprintf(&nameflag, "-_%s=", ctx->i_compiler->c_name) == -1)
666                 nomem();

668         /*
669          * Walk the argument list, translating as we go ..
670          */
671         while (--ctx->i_oldargc > 0) {
672                 char *arg = *++ctx->i_oldargv;
673                 size_t arglen = strlen(arg);

675                 if (*arg == '-') {
676                         arglen--;
677                 } else {
678                         /*
679                          * Discard inline files that gcc doesn't grok
680                          */
681                         if (!in_output && arglen > 3 &&
682                             strcmp(arg + arglen - 3, ".il") == 0)
683                                 continue;

685                         if (!in_output && is_source_file(arg))
597                         if (!in_output && arglen > 2 &&
598                             arg[arglen - 2] == '.' &&
599                             (arg[arglen - 1] == 'S' || arg[arglen - 1] == 's' ||
600                             arg[arglen - 1] == 'c' || arg[arglen - 1] == 'i'))
686                                 c_files++;

688                         /*
689                          * Otherwise, filenames and partial arguments
690                          * are passed through for gcc to chew on.  However,
691                          * output is always discarded for the secondary
692                          * compiler.
693                          */
694                         if ((ctx->i_flags & CW_F_SHADOW) && in_output) {
695                                 newae(ctx->i_ae, discard_file_name(ctx, arg));
696                         } else {
697                                 newae(ctx->i_ae, arg);
698                         }
699                         in_output = 0;
700                         continue;
701                 }

703                 if (ctx->i_flags & CW_F_CXX) {
704                         if (strncmp(arg, "-_g++=", 6) == 0) {
705                                 newae(ctx->i_ae, strchr(arg, '=') + 1);
706                                 continue;
707                         }
708                         if (strncmp(arg, "-compat=", 8) == 0) {
709                                 /* discard -compat=4 and -compat=5 */
710                                 continue;
711                         }
712                         if (strcmp(arg, "-Qoption") == 0) {
713                                 /* discard -Qoption and its two arguments */
714                                 if (ctx->i_oldargc < 3)
715                                         error(arg);
716                                 ctx->i_oldargc -= 2;
717                                 ctx->i_oldargv += 2;
```

```
718                             continue;
719                     }
720                     if (strcmp(arg, "-xwe") == 0) {
721                             /* turn warnings into errors */
722                             newae(ctx->i_ae, "-Werror");
723                             continue;
724                     }
725                     if (strcmp(arg, "-norunpath") == 0) {
726                             /* gcc has no corresponding option */
727                             continue;
728                     }
729                     if (strcmp(arg, "-nolib") == 0) {
730                             /* -nodefaultlibs is on by default */
731                             nolibc = 1;
732                             continue;
733                     }
734 #if defined(__sparc)
735                     if (strcmp(arg, "-cg92") == 0) {
736                             mflag |= xlate_xtb(ctx->i_ae, "v8");
737                             xlate(ctx->i_ae, "super", xchip_tbl);
738                             continue;
739                     }
740 #endif  /* __sparc */
741             }

743             switch ((c = arg[1])) {
744             case '_':
745                     if ((strncmp(arg, nameflag, strlen(nameflag)) == 0) ||
746                         (strncmp(arg, "-_gcc=", 6) == 0) ||
747                         (strncmp(arg, "-_gnu=", 6) == 0)) {
748                             newae(ctx->i_ae, strchr(arg, '=') + 1);
749                     }
750                     break;
751             case '#':
752                     if (arglen == 1) {
753                             newae(ctx->i_ae, "-v");
754                             break;
755                     }
756                     error(arg);
757                     break;
758             case 'f':
759                     if ((strcmp(arg, "-fpic") == 0) ||
760                         (strcmp(arg, "-fPIC") == 0)) {
761                             newae(ctx->i_ae, arg);
762                             break;
763                     }
764                     error(arg);
765                     break;
766             case 'g':
767                     newae(ctx->i_ae, "-gdwarf-2");
768                     break;
769             case 'E':
770                     if (arglen == 1) {
771                             newae(ctx->i_ae, "-xc");
772                             newae(ctx->i_ae, arg);
773                             op = CW_O_PREPROCESS;
774                             nolibc = 1;
775                             break;
776                     }
777                     error(arg);
778                     break;
779             case 'c':
780             case 'S':
781                     if (arglen == 1) {
782                             op = CW_O_COMPILE;
783                             nolibc = 1;
```



```
784                     }
785                     /* FALLTHROUGH */
786             case 'C':
787             case 'H':
788             case 'p':
789                     if (arglen == 1) {
790                             newae(ctx->i_ae, arg);
791                             break;
792                     }
793                     error(arg);
794                     break;
795             case 'A':
796             case 'h':
797             case 'I':
798             case 'i':
799             case 'L':
800             case 'l':
801             case 'R':
802             case 'U':
803             case 'u':
804             case 'w':
805                     newae(ctx->i_ae, arg);
806                     break;
807             case 'o':
808                     seen_o = 1;
809                     if (arglen == 1) {
810                             in_output = 1;
811                             newae(ctx->i_ae, arg);
812                     } else if (ctx->i_flags & CW_F_SHADOW) {
813                             newae(ctx->i_ae, "-o");
814                             newae(ctx->i_ae, discard_file_name(ctx, arg));
815                     } else {
816                             newae(ctx->i_ae, arg);
817                     }
818                     break;
819             case 'D':
820                     newae(ctx->i_ae, arg);
821                     /*
822                      * XXX  Clearly a hack ... do we need _KADB too?
823                      */
824                     if (strcmp(arg, "-D_KERNEL") == 0 ||
825                         strcmp(arg, "-D_BOOT") == 0)
826                             newae(ctx->i_ae, "-ffreestanding");
827                     break;
828             case 'd':
829                     if (arglen == 2) {
830                             if (strcmp(arg, "-dy") == 0) {
831                                     newae(ctx->i_ae, "-Wl,-dy");
832                                     break;
833                             }
834                             if (strcmp(arg, "-dn") == 0) {
835                                     newae(ctx->i_ae, "-Wl,-dn");
836                                     break;
837                             }
838                     }
839                     if (strcmp(arg, "-dalign") == 0) {
840                             /*
841                              * -dalign forces alignment in some cases;
842                              * gcc does not need any flag to do this.
843                              */
844                             break;
845                     }
846                     error(arg);
847                     break;
848             case 'e':
849                     if (strcmp(arg,
```

```
850                             "-erroff=E_EMPTY_TRANSLATION_UNIT") == 0) {
851                                 /*
852                                  * Accept but ignore this -- gcc doesn't
853                                  * seem to complain about empty translation
854                                  * units
855                                  */
856                                 break;
857                         }
858                         /* XX64 -- ignore all -erroff= options, for now */
859                         if (strncmp(arg, "-erroff=", 8) == 0)
860                                 break;
861                         if (strcmp(arg, "-errtags=yes") == 0) {
862                                 warnings(ctx->i_ae);
863                                 break;
864                         }
865                         if (strcmp(arg, "-errwarn=%all") == 0) {
866                                 newae(ctx->i_ae, "-Werror");
867                                 break;
868                         }
869                         error(arg);
870                         break;
871                 case 'G':
872                         newae(ctx->i_ae, "-shared");
873                         nolibc = 1;
874                         break;
875                 case 'k':
876                         if (strcmp(arg, "-keeptmp") == 0) {
877                                 newae(ctx->i_ae, "-save-temps");
878                                 break;
879                         }
880                         error(arg);
881                         break;
882                 case 'm':
883                         if (strcmp(arg, "-mt") == 0) {
884                                 newae(ctx->i_ae, "-D_REENTRANT");
885                                 break;
886                         }
887                         if (strcmp(arg, "-m64") == 0) {
888                                 newae(ctx->i_ae, "-m64");
889 #if defined(__x86)
890                                 newae(ctx->i_ae, "-mtune=opteron");
891 #endif
892                                 mflag |= M64;
893                                 break;
894                         }
895                         if (strcmp(arg, "-m32") == 0) {
896                                 newae(ctx->i_ae, "-m32");
897                                 mflag |= M32;
898                                 break;
899                         }
900                         error(arg);
901                         break;
902                 case 'B':       /* linker options */
903                 case 'M':
904                 case 'z':
905                         {
906                                 char *opt;
907                                 size_t len;
908                                 char *s;

910                                 if (arglen == 1) {
911                                         opt = *++ctx->i_oldargv;
912                                         if (opt == NULL || *opt == '\0')
913                                                 error(arg);
914                                         ctx->i_oldargc--;
915                                 } else {
```



```
916                                         opt = arg + 2;
917                                 }
918                                 len = strlen(opt) + 7;
919                                 if ((s = malloc(len)) == NULL)
920                                         nomem();
921                                 (void) snprintf(s, len, "-Wl,-%c%s", c, opt);
922                                 newae(ctx->i_ae, s);
923                                 free(s);
924                         }
925                         break;
926                 case 'O':
927                         if (arglen == 1) {
928                                 newae(ctx->i_ae, "-O");
929                                 break;
930                         }
931                         error(arg);
932                         break;
933                 case 'P':
934                         /*
935                          * We could do '-E -o filename.i', but that's hard,
936                          * and we don't need it for the case that's triggering
937                          * this addition.  We'll require the user to specify
938                          * -o in the Makefile.  If they don't they'll find out
939                          * in a hurry.
940                          */
941                         newae(ctx->i_ae, "-E");
942                         op = CW_O_PREPROCESS;
943                         nolibc = 1;
944                         break;
945                 case 's':
946                         if (arglen == 1) {
947                                 newae(ctx->i_ae, "-Wl,-s");
948                                 break;
949                         }
950                         error(arg);
951                         break;
952                 case 't':
953                         if (arglen == 1) {
954                                 newae(ctx->i_ae, "-Wl,-t");
955                                 break;
956                         }
957                         error(arg);
958                         break;
959                 case 'V':
960                         if (arglen == 1) {
961                                 ctx->i_flags &= ~CW_F_ECHO;
962                                 newae(ctx->i_ae, "--version");
963                                 break;
964                         }
965                         error(arg);
966                         break;
967                 case 'v':
968                         if (arglen == 1) {
969                                 warnings(ctx->i_ae);
970                                 break;
971                         }
972                         error(arg);
973                         break;
974                 case 'W':
975                         if (strncmp(arg, "-Wp,-xc99", 9) == 0) {
976                                 /*
977                                  * gcc's preprocessor will accept c99
978                                  * regardless, so accept and ignore.
979                                  */
980                                 break;
981                         }
```

```
                        if (strncmp(arg, "-Wa,", 4) == 0 ||
                            strncmp(arg, "-Wp,", 4) == 0 ||
                            strncmp(arg, "-Wl,", 4) == 0) {
                                newae(ctx->i_ae, arg);
                                break;
                        }
                        if (strcmp(arg, "-W0,-noglobal") == 0 ||
                            strcmp(arg, "-W0,-xglobalstatic") == 0) {
                                /*
                                 * gcc doesn't prefix local symbols
                                 * in debug mode, so this is not needed.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-W0,-Lt") == 0) {
                                /*
                                 * Generate tests at the top of loops.
                                 * There is no direct gcc equivalent, ignore.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-W0,-xdbggen=no%usedonly") == 0) {
                                newae(ctx->i_ae,
                                    "-fno-eliminate-unused-debug-symbols");
                                newae(ctx->i_ae,
                                    "-fno-eliminate-unused-debug-types");
                                break;
                        }
                        if (strcmp(arg, "-W2,-xwrap_int") == 0) {
                                /*
                                 * Use the legacy behaviour (pre-SS11)
                                 * for integer wrapping.
                                 * gcc does not need this.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-Wd,-xsafe=unboundsym") == 0) {
                                /*
                                 * Prevents optimizing away checks for
                                 * unbound weak symbol addresses.  gcc does
                                 * not do this, so it's not needed.
                                 */
                                break;
                        }
                        if (strncmp(arg, "-Wc,-xcode=", 11) == 0) {
                                xlate(ctx->i_ae, arg + 11, xcode_tbl);
                                break;
                        }
                        if (strncmp(arg, "-Wc,-Qiselect", 13) == 0) {
                                /*
                                 * Prevents insertion of register symbols.
                                 * gcc doesn't do this, so ignore it.
                                 */
                                break;
                        }
                        if (strcmp(arg, "-Wc,-Qassembler-ounrefsym=0") == 0) {
                                /*
                                 * Prevents optimizing away of static variables.
                                 * gcc does not do this, so it's not needed.
                                 */
                                break;
                        }
#if defined(__x86)
                        if (strcmp(arg, "-Wu,-save_args") == 0) {
                                newae(ctx->i_ae, "-msave-args");
                                break;
```



```
                        }
#endif  /* __x86 */
                        error(arg);
                        break;
                case 'X':
                        if (strcmp(arg, "-Xa") == 0 ||
                            strcmp(arg, "-Xt") == 0) {
                                break;
                        }
                        if (strcmp(arg, "-Xs") == 0) {
                                Xsmode(ctx->i_ae);
                                break;
                        }
                        error(arg);
                        break;
                case 'x':
                        if (arglen == 1)
                                error(arg);
                        switch (arg[2]) {
                        case 'a':
                                if (strncmp(arg, "-xarch=", 7) == 0) {
                                        mflag |= xlate_xtb(ctx->i_ae, arg + 7);
                                        break;
                                }
                                error(arg);
                                break;
                        case 'b':
                                if (strncmp(arg, "-xbuiltin=", 10) == 0) {
                                        if (strcmp(arg + 10, "%all"))
                                                newae(ctx->i_ae, "-fbuiltin");
                                        break;
                                }
                                error(arg);
                                break;
                        case 'C':
                                /* Accept C++ style comments -- ignore */
                                if (strcmp(arg, "-xCC") == 0)
                                        break;
                                error(arg);
                                break;
                        case 'c':
                                if (strncmp(arg, "-xc99=%all", 10) == 0) {
                                        newae(ctx->i_ae, "-std=gnu99");
                                        break;
                                }
                                if (strncmp(arg, "-xc99=%none", 11) == 0) {
                                        newae(ctx->i_ae, "-std=gnu89");
                                        break;
                                }
                                if (strncmp(arg, "-xchip=", 7) == 0) {
                                        xlate(ctx->i_ae, arg + 7, xchip_tbl);
                                        break;
                                }
                                if (strncmp(arg, "-xcode=", 7) == 0) {
                                        xlate(ctx->i_ae, arg + 7, xcode_tbl);
                                        break;
                                }
                                if (strncmp(arg, "-xcrossfile", 11) == 0)
                                        break;
                                error(arg);
                                break;
                        case 'd':
                                if (strncmp(arg, "-xdebugformat=", 14) == 0)
                                        break;
                                error(arg);
                                break;
```

```
1114                        case 'F':
1115                                /*
1116                                 * Compile for mapfile reordering, or unused
1117                                 * section elimination, syntax can be -xF or
1118                                 * more complex, like -xF=%all -- ignore.
1119                                 */
1120                                if (strncmp(arg, "-xF", 3) == 0)
1121                                        break;
1122                                error(arg);
1123                                break;
1124                        case 'i':
1125                                if (strncmp(arg, "-xinline", 8) == 0)
1126                                        /* No inlining; ignore */
1127                                        break;
1128                                if (strcmp(arg, "-xildon") == 0 ||
1129                                    strcmp(arg, "-xildoff") == 0)
1130                                        /* No incremental linking; ignore */
1131                                        break;
1132                                error(arg);
1133                                break;
1134 #if defined(__x86)
1135                        case 'm':
1136                                if (strcmp(arg, "-xmodel=kernel") == 0) {
1137                                        newae(ctx->i_ae, "-ffreestanding");
1138                                        newae(ctx->i_ae, "-mno-red-zone");
1139                                        model = "-mcmodel=kernel";
1140                                        nolibc = 1;
1141                                        break;
1142                                }
1143                                error(arg);
1144                                break;
1145 #endif  /* __x86 */
1146                        case 'O':
1147                                if (strncmp(arg, "-xO", 3) == 0) {
1148                                        size_t len = strlen(arg);
1149                                        char *s = NULL;
1150                                        int c = *(arg + 3);
1151                                        int level;

1153                                        if (len != 4 || !isdigit(c))
1154                                                error(arg);

1156                                        level = atoi(arg + 3);
1157                                        if (level > 5)
1158                                                error(arg);
1159                                        if (level >= 2) {
1160                                                /*
1161                                                 * For gcc-3.4.x at -O2 we
1162                                                 * need to disable optimizations
1163                                                 * that break ON.
1164                                                 */
1165                                                optim_disable(ctx->i_ae, level);
1166                                                /*
1167                                                 * limit -xO3 to -O2 as well.
1168                                                 */
1169                                                level = 2;
1170                                        }
1171                                        if (asprintf(&s, "-O%d", level) == -1)
1172                                                nomem();
1173                                        newae(ctx->i_ae, s);
1174                                        free(s);
1175                                        break;
1176                                }
1177                                error(arg);
1178                                break;
1179                        case 'r':
```



```
1180                                if (strncmp(arg, "-xregs=", 7) == 0) {
1181                                        xlate(ctx->i_ae, arg + 7, xregs_tbl);
1182                                        break;
1183                                }
1184                                error(arg);
1185                                break;
1186                        case 's':
1187                                if (strcmp(arg, "-xs") == 0 ||
1188                                    strcmp(arg, "-xspace") == 0 ||
1189                                    strcmp(arg, "-xstrconst") == 0)
1190                                        break;
1191                                error(arg);
1192                                break;
1193                        case 't':
1194                                if (strncmp(arg, "-xtarget=", 9) == 0) {
1195                                        xlate(ctx->i_ae, arg + 9, xtarget_tbl);
1196                                        break;
1197                                }
1198                                error(arg);
1199                                break;
1200                        case 'e':
1201                        case 'h':
1202                        case 'l':
1203                        default:
1204                                error(arg);
1205                                break;
1206                        }
1207                        break;
1208                case 'Y':
1209                        if (arglen == 1) {
1210                                if ((arg = *++ctx->i_oldargv) == NULL ||
1211                                    *arg == '\0')
1212                                        error("-Y");
1213                                ctx->i_oldargc--;
1214                                arglen = strlen(arg + 1);
1215                        } else {
1216                                arg += 2;
1217                        }
1218                        /* Just ignore -YS,... for now */
1219                        if (strncmp(arg, "S,", 2) == 0)
1220                                break;
1221                        if (strncmp(arg, "l,", 2) == 0) {
1222                                char *s = strdup(arg);
1223                                s[0] = '-';
1224                                s[1] = 'B';
1225                                newae(ctx->i_ae, s);
1226                                free(s);
1227                                break;
1228                        }
1229                        if (strncmp(arg, "I,", 2) == 0) {
1230                                char *s = strdup(arg);
1231                                s[0] = '-';
1232                                s[1] = 'I';
1233                                newae(ctx->i_ae, "-nostdinc");
1234                                newae(ctx->i_ae, s);
1235                                free(s);
1236                                break;
1237                        }
1238                        error(arg);
1239                        break;
1240                case 'Q':
1241                        /*
1242                         * We could map -Qy into -Wl,-Qy etc.
1243                         */
1244                default:
1245                        error(arg);
```

```
1246                        break;
1247                }
1248        }

1250        free(nameflag);

1252        /*
1253         * When compiling multiple source files in a single invocation some
1254         * compilers output objects into the current directory with
1255         * predictable and conventional names.
1256         *
1257         * We prevent any attempt to compile multiple files at once so that
1258         * any such objects created by a shadow can't escape into a later
1259         * link-edit.
1260         */
1261        if (c_files > 1 && op != CW_O_PREPROCESS) {
1262                errx(2, "multiple source files are "
1263                    "allowed only with -E or -P");
1264        }

1266        /*
1267         * Make sure that we do not have any unintended interactions between
1268         * the xarch options passed in and the version of the Studio compiler
1269         * used.
1270         */
1271        if ((mflag & (SS11|SS12)) == (SS11|SS12)) {
1272                errx(2,
1273                    "Conflicting \"-xarch=\" flags (both Studio 11 and 12)\n");
1274        }

1276        switch (mflag) {
1277        case 0:
1278                /* FALLTHROUGH */
1279        case M32:
1280 #if defined(__sparc)
1281                /*
1282                 * Only -m32 is defined and so put in the missing xarch
1283                 * translation.
1284                 */
1285                newae(ctx->i_ae, "-mcpu=v8");
1286                newae(ctx->i_ae, "-mno-v8plus");
1287 #endif
1288                break;
1289        case M64:
1290 #if defined(__sparc)
1291                /*
1292                 * Only -m64 is defined and so put in the missing xarch
1293                 * translation.
1294                 */
1295                newae(ctx->i_ae, "-mcpu=v9");
1296 #endif
1297                break;
1298        case SS12:
1299 #if defined(__sparc)
1300                /* no -m32/-m64 flag used - this is an error for sparc builds */
1301                (void) fprintf(stderr, "No -m32/-m64 flag defined\n");
1302                exit(2);
1303 #endif
1304                break;
1305        case SS11:
1306                /* FALLTHROUGH */
1307        case (SS11|M32):
1308        case (SS11|M64):
1309                break;
1310        case (SS12|M32):
1311 #if defined(__sparc)
```



```
1312                /*
1313                 * Need to add in further 32 bit options because with SS12
1314                 * the xarch=sparcvis option can be applied to 32 or 64
1315                 * bit, and so the translatation table (xtbl) cannot handle
1316                 * that.
1317                 */
1318                newae(ctx->i_ae, "-mv8plus");
1319 #endif
1320                break;
1321        case (SS12|M64):
1322                break;
1323        default:
1324                (void) fprintf(stderr,
1325                    "Incompatible -xarch= and/or -m32/-m64 options used.\n");
1326                exit(2);
1327        }

1329        if (ctx->i_flags & CW_F_SHADOW) {
1330                if (op == CW_O_PREPROCESS)
1331                        exit(0);
1332                else if (op == CW_O_LINK && c_files == 0)
1333                        exit(0);
1334        }

1336        if (model != NULL)
1337                newae(ctx->i_ae, model);
1338        if (!nolibc)
1339                newae(ctx->i_ae, "-lc");
1340        if (!seen_o && (ctx->i_flags & CW_F_SHADOW)) {
1341                newae(ctx->i_ae, "-o");
1342                newae(ctx->i_ae, discard_file_name(ctx, NULL));
1343        }
1344 }
```

*_____unchanged_portion_omitted_*

```
1372 static void
1373 do_cc(cw_ictx_t *ctx)
1374 {
1375        int in_output = 0, seen_o = 0, c_files = 0;
1376        cw_op_t op = CW_O_LINK;
1377        char *nameflag;

1379        if (ctx->i_flags & CW_F_PROG) {
1380                newae(ctx->i_ae, "-V");
1381                return;
1382        }

1384        if (asprintf(&nameflag, "-_%s=", ctx->i_compiler->c_name) == -1)
1385                nomem();

1387        while (--ctx->i_oldargc > 0) {
1388                char *arg = *++ctx->i_oldargv;
1304                size_t arglen = strlen(arg);

1390                if (strncmp(arg, "-_CC=", 5) == 0) {
1391                        newae(ctx->i_ae, strchr(arg, '=') + 1);
1392                        continue;
1393                }

1395                if (*arg != '-') {
1396                        if (!in_output && is_source_file(arg))
1312                        if (!in_output && arglen > 2 &&
1313                            arg[arglen - 2] == '.' &&
1314                            (arg[arglen - 1] == 'S' || arg[arglen - 1] == 's' ||
1315                            arg[arglen - 1] == 'c' || arg[arglen - 1] == 'i'))
1397                                c_files++;
```

```
1399                        if (in_output == 0 || !(ctx->i_flags & CW_F_SHADOW)) {
1400                                newae(ctx->i_ae, arg);
1401                        } else {
1402                                in_output = 0;
1403                                newae(ctx->i_ae, discard_file_name(ctx, arg));
1404                        }
1405                        continue;
1406                }
1407                switch (*(arg + 1)) {
1408                case '_':
1409                        if ((strncmp(arg, nameflag, strlen(nameflag)) == 0) ||
1410                            (strncmp(arg, "-_cc=", 5) == 0) ||
1411                            (strncmp(arg, "-_sun=", 6) == 0)) {
1412                                newae(ctx->i_ae, strchr(arg, '=') + 1);
1413                        }
1414                        break;

1416                case 'V':
1417                        ctx->i_flags &= ~CW_F_ECHO;
1418                        newae(ctx->i_ae, arg);
1419                        break;
1420                case 'o':
1421                        seen_o = 1;
1422                        if (strlen(arg) == 2) {
1423                                in_output = 1;
1424                                newae(ctx->i_ae, arg);
1425                        } else if (ctx->i_flags & CW_F_SHADOW) {
1426                                newae(ctx->i_ae, "-o");
1427                                newae(ctx->i_ae, discard_file_name(ctx, arg));
1428                        } else {
1429                                newae(ctx->i_ae, arg);
1430                        }
1431                        break;
1432                case 'c':
1433                case 'S':
1434                        if (strlen(arg) == 2)
1435                                op = CW_O_COMPILE;
1436                        newae(ctx->i_ae, arg);
1437                        break;
1438                case 'E':
1439                case 'P':
1440                        if (strlen(arg) == 2)
1441                                op = CW_O_PREPROCESS;
1442                /*FALLTHROUGH*/
1443                default:
1444                        newae(ctx->i_ae, arg);
1445                }
1446        }

1448        free(nameflag);

1450        /* See the comment on this same code in do_gcc() */
1451        if (c_files > 1 && op != CW_O_PREPROCESS) {
1452                errx(2, "multiple source files are "
1453                    "allowed only with -E or -P");
1454        }

1456        if (ctx->i_flags & CW_F_SHADOW) {
1457                if (op == CW_O_PREPROCESS)
1458                        exit(0);
1459                else if (op == CW_O_LINK && c_files == 0)
1460                        exit(0);
1461        }

1463        if (!seen_o && (ctx->i_flags & CW_F_SHADOW)) {
```



```
1464                        newae(ctx->i_ae, "-o");
1465                        newae(ctx->i_ae, discard_file_name(ctx, NULL));
1466                }
1467        }
______unchanged_portion_omitted_

1676 static void
1677 cleanup(cw_ictx_t *ctx)
1678 {
1679         DIR *dirp;
1680         struct dirent *dp;
1681         char buf[MAXPATHLEN];

1683         if ((dirp = opendir(ctx->i_tmpdir)) == NULL) {
1684                 if (errno != ENOENT) {
1685                         err(1, "couldn't open temp directory: %s",
1686                             ctx->i_tmpdir);
1687                 } else {
1688                         return;
1689                 }
1690         }

1692         errno = 0;
1693         while ((dp = readdir(dirp)) != NULL) {
1694                 (void) snprintf(buf, MAXPATHLEN, "%s/%s", ctx->i_tmpdir,
1695                     dp->d_name);

1697                 if (strcmp(dp->d_name, ".") == 0 ||
1698                     strcmp(dp->d_name, "..") == 0) {
1616                 if (strncmp(dp->d_name, ".", strlen(dp->d_name)) == 0 ||
1617                     strncmp(dp->d_name, "..", strlen(dp->d_name)) == 0)
1699                         continue;
1700                 }
1701 #endif /* ! codereview */

1703                 if (unlink(buf) == -1)
1704                         err(1, "failed to unlink temp file: %s", dp->d_name);
1705                 errno = 0;
1706         }

1708         if (errno != 0) {
1709                 err(1, "failed to read temporary directory: %s",
1710                     ctx->i_tmpdir);
1711         }

1713         (void) closedir(dirp);
1714         if (rmdir(ctx->i_tmpdir) != 0) {
1715                 err(1, "failed to unlink temporary directory: %s",
1716                     ctx->i_tmpdir);
1717         }
1718 }

1720 int
1721 main(int argc, char **argv)
1722 {
1723         int ch;
1724         cw_compiler_t primary = { NULL, NULL, 0 };
1725         cw_compiler_t shadows[10];
1726         int nshadows = 0;
1727         int ret = 0;
1728         boolean_t do_serial = B_FALSE;
1729         boolean_t do_exec = B_FALSE;
1730         boolean_t vflg = B_FALSE;
1731         boolean_t Cflg = B_FALSE;
1732         boolean_t cflg = B_FALSE;
1733         boolean_t nflg = B_FALSE;
```

```
1734         char *tmpdir;

1736         cw_ictx_t *main_ctx;

1738         static struct option longopts[] = {
1739                 { "compiler", no_argument, NULL, 'c' },
1740                 { "noecho", no_argument, NULL, 'n' },
1741                 { "primary", required_argument, NULL, 'p' },
1742                 { "shadow", required_argument, NULL, 's' },
1743                 { "versions", no_argument, NULL, 'v' },
1744                 { NULL, 0, NULL, 0 },
1745         };


1748         if ((main_ctx = newictx()) == NULL)
1749                 nomem();

1751         while ((ch = getopt_long(argc, argv, "C", longopts, NULL)) != -1) {
1752                 switch (ch) {
1753                 case 'c':
1754                         cflg = B_TRUE;
1755                         break;
1756                 case 'C':
1757                         Cflg = B_TRUE;
1758                         break;
1759                 case 'n':
1760                         nflg = B_TRUE;
1761                         break;
1762                 case 'p':
1763                         if (primary.c_path != NULL) {
1764                                 warnx("Only one primary compiler may "
1765                                     "be specified");
1766                                 usage();
1767                         }

1769                         parse_compiler(optarg, &primary);
1770                         break;
1771                 case 's':
1772                         if (nshadows >= 10)
1773                                 errx(1, "May only use 10 shadows at "
1774                                     "the moment");
1775                         parse_compiler(optarg, &shadows[nshadows]);
1776                         nshadows++;
1777                         break;
1778                 case 'v':
1779                         vflg = B_TRUE;
1780                         break;
1781                 default:
1782                         (void) fprintf(stderr, "Did you forget '--'?\n");
1783                         usage();
1784                 }
1785         }

1787         if (primary.c_path == NULL) {
1788                 warnx("A primary compiler must be specified");
1789                 usage();
1790         }

1792         do_serial = (getenv("CW_SHADOW_SERIAL") == NULL) ? B_FALSE : B_TRUE;
1793         do_exec = (getenv("CW_NO_EXEC") == NULL) ? B_TRUE : B_FALSE;

1795         /* Leave room for argv[0] */
1796         argc -= (optind - 1);
1797         argv += (optind - 1);

1799         main_ctx->i_oldargc = argc;
```



```
1800         main_ctx->i_oldargv = argv;
1801         main_ctx->i_flags = CW_F_XLATE;
1802         if (nflg == 0)
1803                 main_ctx->i_flags |= CW_F_ECHO;
1804         if (do_exec)
1805                 main_ctx->i_flags |= CW_F_EXEC;
1806         if (Cflg)
1807                 main_ctx->i_flags |= CW_F_CXX;
1808         main_ctx->i_compiler = &primary;

1810         if (cflg) {
1811                 (void) fputs(primary.c_path, stdout);
1812         }

1814         if (vflg) {
1815                 (void) printf("cw version %s\n", CW_VERSION);
1816                 (void) fflush(stdout);
1817                 main_ctx->i_flags &= ~CW_F_ECHO;
1818                 main_ctx->i_flags |= CW_F_PROG | CW_F_EXEC;
1819                 do_serial = 1;
1820         }

1822         tmpdir = getenv("TMPDIR");
1823         if (tmpdir == NULL)
1824                 tmpdir = "/tmp";

1826         if (asprintf(&main_ctx->i_tmpdir, "%s/cw.XXXXXX", tmpdir) == -1)
1827                 nomem();

1829         if ((main_ctx->i_tmpdir = mkdtemp(main_ctx->i_tmpdir)) == NULL)
1830                 errx(1, "failed to create temporary directory");

1832         ret |= exec_ctx(main_ctx, do_serial);

1834         for (int i = 0; i < nshadows; i++) {
1835                 int r;
1836                 cw_ictx_t *shadow_ctx;

1838                 if ((shadow_ctx = newictx()) == NULL)
1839                         nomem();

1841                 (void) memcpy(shadow_ctx, main_ctx, sizeof (cw_ictx_t));

1843                 shadow_ctx->i_flags |= CW_F_SHADOW;
1844                 shadow_ctx->i_compiler = &shadows[i];

1846                 r = exec_ctx(shadow_ctx, do_serial);
1847                 if (r == 0) {
1848                         shadow_ctx->i_next = main_ctx->i_next;
1849                         main_ctx->i_next = shadow_ctx;
1850                 }
1851                 ret |= r;
1852         }

1854         if (!do_serial) {
1855                 cw_ictx_t *next = main_ctx;
1856                 while (next != NULL) {
1857                         cw_ictx_t *toreap = next;
1858                         next = next->i_next;
1859                         ret |= reap(toreap);
1860                 }
1861         }

1863         cleanup(main_ctx);
1864         return (ret);
1865 }
```