```
**********************************************************
   17793 Sun Feb 22 13:15:32 2015
new/usr/src/tools/aw/aw.c
switch aw & cw to target ARMv7
**********************************************************
______unchanged_portion_omitted_

480 int
481 main(int argc, char *argv[])
482 {
483         struct aelist *cpp = NULL;
484         struct aelist *m4 = NULL;
485         struct aelist *as = newael();
486         char **asargv;
487         char *outfile = NULL;
488         char *srcfile = NULL;
489         const char *dir, *cmd;
490         static char as_pgm[MAXPATHLEN];
491         static char as64_pgm[MAXPATHLEN];
492         static char m4_pgm[MAXPATHLEN];
493         static char m4_cmdefs[MAXPATHLEN];
494         static char cpp_pgm[MAXPATHLEN];
495         int as64 = 0;
496         int code;

498         if ((progname = strrchr(argv[0], '/')) == NULL)
499                 progname = argv[0];
500         else
501                 progname++;

503         /*
504          * Helpful when debugging, or when changing tool versions..
505          */
506         cmd = dir = NULL;
507         if ((cmd = getenv("AW_" AW_TARGET "_AS")) == NULL)
508                 cmd = getenv("AW_AS");
509         if ((dir = getenv("AW_" AW_TARGET "_AS_DIR")) == NULL)
510                 dir = getenv("AW_AS_DIR");

512         if (cmd != NULL) {
513                 (void) strlcpy(as_pgm, cmd, sizeof (as_pgm));
514         } else {
515                 if (dir == NULL)
516                         dir = DEFAULT_AS_DIR;
517                 (void) snprintf(as_pgm, sizeof (as_pgm), "%s/gas", dir);
518         }

520         cmd = dir = NULL;
521         if ((cmd = getenv("AW_" AW_TARGET "_AS64")) == NULL)
522                 cmd = getenv("AW_AS64");
523         if ((dir = getenv("AW_" AW_TARGET "_AS64_DIR")) == NULL)
524                 dir = getenv("AW_AS64_DIR");

526         if (cmd != NULL) {
527                 (void) strlcpy(as64_pgm, cmd, sizeof (as64_pgm));
528         } else {
529                 if (dir == NULL)
530                         dir = DEFAULT_AS64_DIR;
531                 (void) snprintf(as64_pgm, sizeof (as_pgm), "%s/gas", dir);
532         }

534         if ((cmd = getenv("AW_M4")) != NULL)
535                 strlcpy(m4_pgm, cmd, sizeof (m4_pgm));
536         else {
537                 if ((dir = getenv("AW_M4_DIR")) == NULL)
538                         dir = DEFAULT_M4_DIR;   /* /usr/ccs/bin */
```



```
539                 (void) snprintf(m4_pgm, sizeof (m4_pgm), "%s/m4", dir);
540         }

542         if ((cmd = getenv("AW_M4LIB")) != NULL)
543                 strlcpy(m4_cmdefs, cmd, sizeof (m4_cmdefs));
544         else {
545                 if ((dir = getenv("AW_M4LIB_DIR")) == NULL)
546                         dir = DEFAULT_M4LIB_DIR;        /* /usr/ccs/lib */
547                 (void) snprintf(m4_cmdefs, sizeof (m4_cmdefs),
548                     "%s/cm4defs", dir);
549         }

551         if ((cmd = getenv("AW_CPP")) != NULL)
552                 strlcpy(cpp_pgm, cmd, sizeof (cpp_pgm));
553         else {
554                 if ((dir = getenv("AW_CPP_DIR")) == NULL)
555                         dir = DEFAULT_CPP_DIR;  /* /usr/ccs/lib */
556                 (void) snprintf(cpp_pgm, sizeof (cpp_pgm), "%s/cpp", dir);
557         }

559         newae(as, as_pgm);
560         newae(as, "--warn");
561         newae(as, "--fatal-warnings");
562         newae(as, "--traditional-format");

564         /*
565          * Walk the argument list, translating as we go ..
566          */
567         while (--argc > 0) {
568                 char *arg;
569                 int arglen;

571                 arg = *++argv;
572                 arglen = strlen(arg);

574                 if (*arg != '-') {
575                         char *filename;

577                         /*
578                          * filenames ending in '.s' are taken to be
579                          * assembler files, and provide the default
580                          * basename of the output file.
581                          *
582                          * other files are passed through to the
583                          * preprocessor, if present, or to gas if not.
584                          */
585                         filename = arg;
586                         if ((arglen > 2) &&
587                             ((strcmp(arg + arglen - 2, ".s") == 0) ||
588                             (strcmp(arg + arglen - 2, ".S") == 0))) {
589                                 /*
590                                  * Though 'as' allows multiple assembler
591                                  * files to be processed in one invocation
592                                  * of the assembler, ON only processes one
593                                  * file at a time, which makes things a lot
594                                  * simpler!
595                                  */
596                                 if (srcfile == NULL)
597                                         srcfile = arg;
598                                 else
599                                         return (usage(
600                                             "one assembler file at a time"));

602                                 /*
603                                  * If we haven't seen a -o option yet,
604                                  * default the output to the basename
```

```
605                              * of the input, substituting a .o on the end
606                              */
607                             if (outfile == NULL) {
608                                     char *argcopy;

610                                     argcopy = strdup(arg);
611                                     argcopy[arglen - 1] = 'o';

613                                     if ((outfile = strrchr(
614                                         argcopy, '/')) == NULL)
615                                             outfile = argcopy;
616                                     else
617                                             outfile++;
618                             }
619                     }
620                     if (cpp)
621                             newae(cpp, filename);
622                     else if (m4)
623                             newae(m4, filename);
624                     else
625                             newae(as, filename);
626                     continue;
627             } else
628                     arglen--;

630             switch (arg[1]) {
631             case 'K':
632                     /*
633                      * -K pic
634                      * -K PIC
635                      */
636                     if (arglen == 1) {
637                             if ((arg = *++argv) == NULL || *arg == '\0')
638                                     return (usage("malformed -K"));
639                             argc--;
640                     } else {
641                             arg += 2;
642                     }
643                     if (strcmp(arg, "PIC") != 0 && strcmp(arg, "pic") != 0)
644                             return (usage("malformed -K"));
645                     break;          /* just ignore -Kpic for gcc */
646             case 'Q':
647                     if (strcmp(arg, "-Qn") == 0)
648                             break;
649                     /*FALLTHROUGH*/
650             case 'b':
651             case 's':
652             case 'T':
653                     /*
654                      * -b   Extra symbol table for source browser ..
655                      *      not relevant to gas, thus should error.
656                      * -s   Put stabs in .stabs section not stabs.excl
657                      *      not clear if there's an equivalent
658                      * -T   4.x migration option
659                      */
660             default:
661                     return (error(arg));
662             case 'x':
663                     /*
664                      * Accept -xarch special case to invoke alternate
665                      * assemblers or assembler flags for different
666                      * architectures.
667                      */
668                     if (strcmp(arg, "-xarch=amd64") == 0 ||
669                         strcmp(arg, "-xarch=generic64") == 0) {
670                             as64++;
```



```
671                             fixae_arg(as->ael_head, as64_pgm);
672                             break;
673                     }
674                     /*
675                      * XX64: Is this useful to gas?
676                      */
677                     if (strcmp(arg, "-xmodel=kernel") == 0)
678                             break;

680                     /*
681                      * -xF  Generates performance analysis data
682                      *      no equivalent
683                      */
684                     return (error(arg));
685             case 'V':
686                     newae(as, arg);
687                     break;
688             case '#':
689                     verbose++;
690                     break;
691             case 'L':
692                     newae(as, "--keep-locals");
693                     break;
694             case 'n':
695                     newae(as, "--no-warn");
696                     break;
697             case 'o':
698                     if (arglen != 1)
699                             return (usage("bad -o flag"));
700                     if ((arg = *++argv) == NULL || *arg == '\0')
701                             return (usage("bad -o flag"));
702                     outfile = arg;
703                     argc--;
704                     arglen = strlen(arg + 1);
705                     break;
706             case 'm':
707                     if (cpp)
708                             return (usage("-m conflicts with -P"));
709                     if (m4 == NULL) {
710                             m4 = newael();
711                             newae(m4, m4_pgm);
712                             newae(m4, m4_cmdefs);
713                     }
714                     break;
715             case 'P':
716                     if (m4)
717                             return (usage("-P conflicts with -m"));
718                     if (cpp == NULL) {
719                             cpp = newael();
720                             newae(cpp, cpp_pgm);
721                             newae(cpp, "-D__GNUC_AS__");
722                     }
723                     break;
724             case 'D':
725             case 'U':
726                     if (cpp)
727                             newae(cpp, arg);
728                     else if (m4)
729                             newae(m4, arg);
730                     else
731                             newae(as, arg);
732                     break;
733             case 'I':
734                     if (cpp)
735                             newae(cpp, arg);
736                     else
```

```
737                                 newae(as, arg);
738                         break;
739                 case '-':       /* a gas-specific option */
740                         newae(as, arg);
741                         break;
742                 }
743         }

745 #if defined(AW_TARGET_i386)
746         if (as64)
747                 newae(as, "--64");
748         else
749                 newae(as, "--32");
750 #endif



753         /*
754          * gas for 32-bit arm defaults to a much older version of the arm
755          * architecture than we can really support. Because of that, we instead
756          * opt to make sure that we set the minimum architecture to armv6k, the
757          * minimum of what we actually support.
758          */
759 #if defined(AW_TARGET_arm)
760         if (as64) {
761                 return (error("no 64-bit aw target for arm"));
762         } else {
763                 newae(as, "-march=armv7-a");
764                 newae(as, "-mfpu=vfpv3-d16");
763                 newae(as, "-march=armv6k");
764                 newae(as, "-mfpu=vfpv2");
765                 newae(as, "-mfloat-abi=hard");
766         }
767 #endif

769         if (srcfile == NULL)
770                 return (usage("no source file(s) specified"));
771         if (outfile == NULL)
772                 outfile = "a.out";
773         newae(as, "-o");
774         newae(as, outfile);

776         asargv = aeltoargv(as);
777         if (cpp) {
778 #if defined(AW_TARGET_sparc)
779                 newae(cpp, "-Dsparc");
780                 newae(cpp, "-D__sparc");
781                 if (as64)
782                         newae(cpp, "-D__sparcv9");
783                 else
784                         newae(cpp, "-D__sparcv8");
785 #elif defined(AW_TARGET_i386)
786                 if (as64) {
787                         newae(cpp, "-D__x86_64");
788                         newae(cpp, "-D__amd64");
789                 } else {
790                         newae(cpp, "-Di386");
791                         newae(cpp, "-D__i386");
792                 }
793 #elif defined(AW_TARGET_arm)
794                 newae(cpp, "-Darm");
795                 newae(cpp, "-D__arm");
796 #else
797 #error  "need isa-dependent defines"
798 #endif
799                 code = pipeline(aeltoargv(cpp), asargv);
800         } else if (m4)
```



```
801                 code = pipeline(aeltoargv(m4), asargv);
802         else {
803                 /*
804                  * XXX  should arrange to fork/exec so that we
805                  *      can unlink the output file if errors are
806                  *      detected..
807                  */
808                 (void) execvp(asargv[0], asargv);
809                 perror("execvp");
810                 (void) fprintf(stderr, "%s: couldn't run %s\n",
811                     progname, asargv[0]);
812                 code = 7;
813         }
814         if (code != 0)
815                 (void) unlink(outfile);
816         return (code);
817 }
```

unchanged_portion_omitted_

****
46611 Sun Feb 22 13:15:32 2015
new/usr/src/tools/cw/cw.c
switch aw & cw to target ARMv7
****
_____***unchanged_portion_omitted_***

```
 388 /*
 389  * The translation table for the -xarch= flag used in the Studio compilers.
 390  */
 391 static const xarch_table_t xtbl[] = {
 392 #if defined(CW_TARGET_i386)
 393         { "generic",    SS11 },
 394         { "generic64",  (SS11|M64), { "-m64", "-mtune=opteron" } },
 395         { "amd64",      (SS11|M64), { "-m64", "-mtune=opteron" } },
 396         { "386",        SS11,   { "-march=i386" } },
 397         { "pentium_pro", SS11,  { "-march=pentiumpro" } },
 398         { "sse",        SS11, { "-msse", "-mfpmath=sse" } },
 399         { "sse2",       SS11, { "-msse2", "-mfpmath=sse" } },
 400 #elif defined(CW_TARGET_sparc)
 401         { "generic",    (SS11|M32), { "-m32", "-mcpu=v8" } },
 402         { "generic64",  (SS11|M64), { "-m64", "-mcpu=v9" } },
 403         { "v8",         (SS11|M32), { "-m32", "-mcpu=v8", "-mno-v8plus" } },
 404         { "v8plus",     (SS11|M32), { "-m32", "-mcpu=v9", "-mv8plus" } },
 405         { "v8plusa",    (SS11|M32), { "-m32", "-mcpu=ultrasparc", "-mv8plus",
 406                         "-mvis" } },
 407         { "v8plusb",    (SS11|M32), { "-m32", "-mcpu=ultrasparc3", "-mv8plus",
 408                         "-mvis" } },
 409         { "v9",         (SS11|M64), { "-m64", "-mcpu=v9" } },
 410         { "v9a",        (SS11|M64), { "-m64", "-mcpu=ultrasparc", "-mvis" } },
 411         { "v9b",        (SS11|M64), { "-m64", "-mcpu=ultrasparc3", "-mvis" } },
 412         { "sparc",      SS12, { "-mcpu=v9", "-mv8plus" } },
 413         { "sparcvis",   SS12, { "-mcpu=ultrasparc", "-mvis" } },
 414         { "sparcvis2",  SS12, { "-mcpu=ultrasparc3", "-mvis" } }
 415 #elif defined(CW_TARGET_arm)
 416         { "generic",    SS12, { "-march=armv7-a", "-mfpu=vfpv3-d16", "-mhard-flo
 416         { "generic",    SS12, { "-march=armv6", "-mfpu=vfp", "-mhard-float" } }
 417 #else
 418 #error Unknown CW_TARGET
 419 #endif
 420 };
```

_____***unchanged_portion_omitted_***