NAME

asl - cross assembler for microprocessors and -controllers

SYNTAX

asl [option(s)] file(s) [option(s)] file(s) ...

DESCRIPTION

AS is a cross assembler that can be used to write assembler programs for a variety of different microprocessors and -controllers. **asl** is the UNIX/C implementation of AS. A complete description of AS is far beyond the scope of this manual page, which is only intended as a quick reference for AS's command line interface. For a more detailed introduction into the usage of AS, see the user's manual.

COMMAND-LINE PARAMETERS

Every argument that starts with a slash (/), plus (+) or minus (-) sign is regarded as a command-line parameter. Some command-line parameters take an argument as additional parameter. Every argument that is not recognized as a command-line parameter is regarded as an assembler source file to be assembled. A source file name that does not have an extension is automatically extended with the default extension '.asm'. Options are turned on with a parameter starting with a slash (/) or minus sign (-), whereas a parameter with a leading plus sign (+) turns an option off. In the following list, all options will be shown in the form that is needed to change the default behaviour, which might be a plus or minus sign, depening on wether the option is on or off by default.

asl accepts the following command-line parameters:

-A

Change the data structure that is internally used to store the symbol table. By default, AS uses binary trees to store macro and symbol definitions. Turning this option on will change this to AVL-balanced trees. Depending on the ratio of symbol entries and lookups, this might speed up assembly. Using AVL-balanced trees helps also reducing the stack usage, which is however irrelevant for the C version of AS.

-a

Instruct AS to write out the shared symbol definitions in a format suitable for including into an AS assembler program. The file's name is constructed by replacing the source file's extension with '.inc'. See the user manual for more information about symbol sharing.

-cpu <name>

Set the target processor to <name>. Use this option if the source file does not contain a CPU statement.

-alias <new name=old name>

Define a CPU alias. An alias is a name that can be used as an argument to the CPU pseudo-instruction just like an intrinsic CPU type. Its usage will set the same target as the old name, however

the predefined symbols MOMCPU and MOMCPUNAME will be set to the new name. This option is primarily useful for adding a new member to a processor family that has the same core, but is different in its internal peripherals, thus allowing to distinguish between them.

-C

Add a cross reference table to the assembler listing. A cross reference table lists all symbols that have been referenced at least once during assembly, including the source line number(s) and count of every reference. This option only makes sense when the generation of an assembly listing has been turned on via the **-L** or **-l** parameters.

-c

Instruct AS to write out the shared symbol definitions in a format suitable for including into a C program. The file's name is constructed by replacing the source file's extension with '.h'. See the user manual for more information about symbol sharing.

-compmode

Enable the 'compatibility mode', which aims at making the behaviour of AS more like the respective 'original assembler' in some situations. See the manual for target-specific details.

-D <name[=value]>[,...]

Pre-define symbols. Predefined symbols are entered into the global symbol table prior to assembly of the source file(s). If no value is given for a symbol, it is assigned the integer value 1. Value expressions may contain arithmetic expressions as described in the user manual, but they may not reference any other predefined or internal symbols.

-E [file]

Force AS to send warning and error messages to **file** rather than to the standard error output. The file names !0 to !4 are used to refer to the standard input, output, error, auxilliary, and printer channels predefined by the operating system (on some systems, some of these handles might not exist). If the [file] specification is omitted, a name is constructed by replacing the source file's extension with '.log'.

+G Supress code generation, reducing the functionality of AS to macro preprocessing.

-g [MAP|Atmel|NoICE]

Instruct AS to write an additional file containing debug information. This information covers the symbol table and the relation between source line numbers and machine addresses. The argument specifies whether debug info shall be written in AS's own MAP format, the object format for Atmel's AVR tools, or a command file suitable for John Hartman's NoICE. If no argument is given, MAP will be chosen. The file's name is constructed by replacing the source file's extension with '.map', '.obj', or '.noi' respectively.

-gnuerrors

Output errors and their location in the source code in a format similar to the GNU C compiler, thus making it easier to integrate AS into environments designed for this format. Note that in contrast to the standard format used by AS, locations inside macros are not reported!

-maxerrors <num>

Force AS to terminate assembly after the given number of errors occured.

-maxinclevel <num>

Limit the maximum nesting level of include statements to the given number. By default, this limit is set to 200.

-werror

Force AS to treat warnings as errors.

-h

Force AS to print all hexadecimal constants with lowercase letters, rather than with uppercase letters A..F which is the default.

-listradix <2..36>

Set the number system to be used in the listing. By default, all integer values are printed in hex, but an arbitrary base between 2 and 36 may be used via this switch.

-splitbyte [character]

Use split-byte display of numbers in the listing, e.g. to obtain split octal notation.

-i <path[:path...]>

Add new entries to the list of paths that are searched for include files. New entries are prepended to the current include path list, so if multiple paths are given with one command-line parameter, they will be entered into the path list in reverse order.

-I

Add an include file list to the assembly listing. An include file list contains all files that have been included while assembling the source files, including multiple and nested inclusion. Nesting of inclusion is identified by different indention. This option only makes sense when the generation of an assembly listing has been turned on via the **-L** or **-l** parameters.

-L

Turn on generation of an assembly listing and send it to a file whose name is constructed by replacing the source file's extension with '.lst'.

-l

Turn on generation of an assembly listing and send it to the console.

-olist <file name>

Override the default path and file name for the file the listing is written to.

-M

Turn on generation of a macro definition file. A macro definition file is a file that contains all macro definitions that have been detected during assembly, in a format suitable for an inclusion into another file. The macro definition file's name is constructed by replacing the source file's extension with '.mac'.

-n

Force AS to extend all error and warning messages with their internal error resp. warning number.

-noicemask [mask]

By default, AS will only write symbols to the NoICE debug info that are located in the code segment. By changing this binary mask, more and/or different masks may be chosen. The assignment is bit 1=code, 2=data, 3=idata, 4=xdata, 5=ydata, 6=bitdata, 7=io, 8=reg, 9=romdata . Negating this option reverts the mask to code-only.

-o <name>

Tell AS to write the code output to a file <name>. By default, the code output file's name is constructed by replacing the source file's extension with '.p'. If multiple files are assembled with one command, this parameter may also be given multiply. If there are less output specifications than source file names, AS will switch back to the default naming convention after all output specifications have been used up.

-olist <name>

Tell AS to write the assembly lsting to a file <name>, instead of a file with the standard name in the source file's directory. This option only makes sense when an assembly listing has been requested via the '-L' option.

-shareout <name>

Tell AS to write shared symbol definitions to a file <name>, instead of constructing the name from the source file's name. See the user manual for more information about symbol sharing.

-P

Turn on generation of a macro output file. A macro output file contains the intermediate source code that remains after macro expansion and conditional assembly. The macro output file's name is constructed by replacing the source file's extension with '.i'.

-p

Instruct AS to write out the shared symbol definitions in a format suitable for including into a Pascal or Modula-2 program. The file's name is constructed by replacing the source file's extension with '.inc'. See the user manual for more information about symbol sharing.

-q or -quiet

Turn on silent assembly mode. In silent compilation mode, AS will not do any console output except for warning and error messages.

-r [pass number]

Tell AS to output warnings when a situation appears in a source file that forces another pass of assembly. Such situations either take place when a symbol is undefined in the first pass or a symbol's value has changed compared to the previous pass. This option is useful to track down sources of excessive multi-passing, but be aware that it might yield a fairly large number of warnings, especially in the first pass. Optionally, a pass number may be added to this option to inhibit output until a certain pass is reached.

-relaxed

Enable the relaxed mode by default, which allows all syntax variants for hexadecimal/binary/ocatal constants (Intel, Motorola, Hitachi, C), rahter than only the target-specific one.

-s

Add a section list to the assembly listing. A section list contains all sections that have been defined in the source files, marking their nesting level by different levels of indentation. This option only makes sense when the generation of an assembly listing has been turned on via the **-L** or **-l** parameters.

-shareout <file>

Override the default path and file name for the file containing shared symbols.

-t <mask>

Turn on or off parts of the assembly listing that have no individual command line parameter. AS internally keeps an integer value whose bits represent certain components of the listing. A positive command line parameter (-t or /t) sets the bits set in <mask>, whereas a negated parameter (+t) resets the bits set in <mask>. The individual bits have the following meaning: bit 0 = source lines and generated machine code, bit 1 = symbol table, bit 2 = macro table, bit 3 = function table, bit 4 = line numbering.

-u

Tell AS to do additional bookkeeping about which address ranges have been used by the assembled program. This option enables the detection of overlapping memory usage. If an assembly listing has been turned on via the **-L** or **-l** parameters, it will also contain a list of all used memory areas.

-U

Force AS to operate in case-sensitive mode. By default, names of symbols, macros, user-defined functions and sections are treated in a case-insensitive manner.

-w

Suppress output of warnings.

-x

Turn on extended error reporting. With extended error reporting, several error and warning messages will also print the item that created the message, e.g. the name of an unknown instruction. When this option is given twice, the erroneous source line is additionally printed.

-warnranges

Only warn about integer constants exceedings the given data type's range, instead of issuing an error.

PRESETTING PARAMETERS

Parameters need not neccessarily be given in the command line itself. Before processing of command line parameters starts, AS will look if the **ASCMD** environment variable is defined. If it exists, its contents will be treated as additional command line parameters whose syntax is absolutely equal to normal command line parameters. An exception is made if the variable's contents start with a '@' sign; in such a case, the string after the '@' sign is treated as the name of a file that contains the options. Such a file (also called a 'key file') has the advantage that it allows the options to be written in different lines, and it does not have a size limit. Some operating systems (like MS-DOS) have a length limit on command lines and environment variable contents, so the key file may be your only option if you have a lot of lengthy parameters for AS.

As parameters given in the **ASCMD** environment variable or a key file are processed prior to the command line itself, and can therefore be overridden by command line parameters.

NATIONAL LANGUAGE SUPPORT

AS supports the needs of different languages and countries in the sense that it will try to adapt to the language and date/time formats valid for the current environment. Upon startup, the COUNTRY setting made in the CONFIG.SYS file is queried for DOS and OS/2 implementations of AS. For UNIX systems, AS tries to read the LC_TIME resp. LC_MESSAGES environment variables to find out the correct format for date/time outputs resp. the local language. If this fails, the LC_ALL and finally LANG variables are probed. If none of these environment variables points to a specific local environment resp. contains a locale specification unknown to AS, the standard english/C locale is used.

The messages AS can output in different languages are stored in separate

files with the extension '.msg' . AS will search for these files in the following directories:

- The current directory

- The directory the executable of AS was loaded from (only on DOS platforms version >=3.0 or if path was explicitly specified)

- The directory specified in the AS_MSGPATH environment variable resp. the directories listed in the PATH environment variable if AS_MSGPATH does not exist.

- The LIBDIR directory set at compile time from the Makefile.

RETURN CODES

Depending on the assembly's turnout, **asl** will generate different return codes:

0 No errors, warnings might have occured.

1 No command line parameters were given, AS printed a short list of possible command line parameters and terminated thereafter.

2 Errors occured during assembly of at least one source file, no code file was generated for the corresponding source file(s).

3 A fatal error occured during assembly that forced immediate program termination. Code files may be generated, but are probably unusuable.

4 Program termination already occured during initialization. This might be either due to a incorrect command line parameter or an error during loading the program's overlay file (only possible on MS-DOS).

255 During initialization, an internal error happened that should not occur. If the reaction is reproducable, note down the situation and send a bug report to the author.

EXAMPLES

To assemble a source file **file1.asm** without any additional bells and whistles, use:

asl file1

which will create an output file **file1.p** given that there are no errors. If you additionally want a listing and rename the output file to **a.out**, use

asl -L file1 -o a.out

To make the listing as comprehensive as possible and to get more detailed error messages, use:

asl -LuCIs -t 16 -nx file1

TIPS

calling AS without any parameters will print a short help containing all command line parameters and implemented target processors, while calling

with command line parameters but without any source file name will result in AS asking for a source file name on the console.

SEE ALSO

plist(1), pbind(1), p2hex(1), p2bin(1)

HISTORY

AS originally appeared as a DOS program in 1989, written in Borland-Pascal, and was ported to C and UNIX in 1996.

BUGS

There are too many options.

Command line interpreters of some operating systems reserve some characters for their own use, so it might be necessary to give command line parameters with certain tricks (e.g., with the help of escape characters).

AUTHOR(S)

Alfred Arnold (alfred@ccac.rwth-aachen.de), and a few others...