Using the Assembler Command Line C

This appendix is organized into the following secitons:

Assembler Command Line	*page 59*
Assembler Command Line Options	*page 60*
Disassembling Object Code	*page 63*

Assembler Command Line

You invoke the assembler command line as follows:

```
as [options] [inputfile] ...
```

Note – The language drivers (such as *cc* and *f77*) invoke the assembler command line with the `fbe` command. You can use either the `as` or `fbe` command to invoke the assembler command line.

The `as` command translates the assembly language source files, *inputfile*, into an executable object file, *objfile*. The SPARC assembler recognizes the filename argument *hyphen* (-) as the standard input. It accepts more than one file name on the command line. The input file is the concatenation of all the specified files. If an invalid option is given or the command line contains a syntax error, the SPARC assembler prints the error (including a synopsis of the command line syntax and options) to standard error output, and then terminates.

C

The SPARC assembler supports macros, `#include` files, and symbolic substitution through use of the C preprocessor *cpp*. The assembler invokes the preprocessor before assembly begins if it has been specified from the command line as an option. (See the `-P` option.)

Assembler Command Line Options

`-b`

This option generates extra symbol table information for the source code browser.

- If the `as` command line option `-P` is set, the `cpp` preprocessor also collects browser information.
- If the `as` command line option `-m` is set, this option is ignored as the `m4` macro processor does not generate browser data.

For more information about the SPARCworks SourceBrowser, see the *Browsing Source Code* manual.

`-Dname`
`-Dname=def`

When the `-P` option is in effect, these options are passed to the `cpp` preprocessor without interpretation by the `as` command; otherwise, they are ignored.

`-Ipath`

When the `-P` option is in effect, this option is passed to the `cpp` preprocessor without interpretation by the `as` command; otherwise, it is ignored.

`-K PIC`

This option generates position-independent code. This option has the same functionality as the `-k` option under the SunOS 4.1 SPARC assembler.

Note – `-K PIC` and `-K pic` are equivalent.

`-L`

Saves all symbols, including temporary labels that are normally discarded to save space, in the ELF symbol table.

`-m`

This option runs `m4` macro preprocessing on input. The `m4` preprocessor is more powerful than the C preprocessor (invoked by the `-P` option), so it is more useful for complex preprocessing. See the `m4(1)` man page for more information about the `m4` macro-processor.

`-n`

Suppress all warnings while assembling.

`-o outfile`

Takes the next argument as the name of the output file to be produced. By default, the .s suffix, if present, is removed from the input file and the .o suffix is appended to form the ouput file name.

`-P`

Run *cpp*, the C preprocessor, on the files being assembled. The preprocessor is run separately on each input file, not on their concatenation. The preprocessor output is passed to the assembler.

`-Q{y|n}`

This option produces the "assembler version" information in the comment section of the output object file if the `y` option is specified; if the `n` option is specified, the information is suppressed.

`-q`

This option causes the assembler to perform a quick assembly. Many error-checks are not performed when `-q` is specified.

Note – This option disables many error checks. It is recommended that you do *not* use this option to assemble handwritten assembly language.

`-S[a|C]`

Produces a disassembly of the emitted code to the standard output.

- Adding the character *a* to the option appends a comment line to each assembly code which indicates its relative address in its own section.
- Adding the character `"C"` to the option prevents comment lines from appearing in the output.

`-s`

This option places all stabs in the "`.stabs`" section. By default, stabs are placed in "`stabs.excl`" sections, which are stripped out by the static linker `ld` during final execution. When the `-s` option is used, stabs remain in the final executable because ".stab" sections are not stripped out by the static linker `ld`.

`-T`

This is a migration option for SunOS 4.1 assembly files to be assembled on SunOS 5.*x* systems. With this option, the symbol names in SunOS 4.1 assembly files will be interpreted as SunOS 5.*x* symbol names. This option can be used in conjunction with the –S option to convert SunOS 4.1 assembly files to their corresponding SunOS 5.*x* versions.

`-Uname`

When the `-P` option is in effect, this option is passed to the `cpp` preprocessor without interpretation by the `as` command; otherwise, it is ignored.

`-V`

This option writes the version information on the standard error output.

`-xarch=v7`

This option instructs the assembler to accept instructions defined in the SPARC version 7 (V7) architecture. The resulting object code is in ELF format.

`-xarch=v8`

This option instructs the assembler to accept instructions defined in the SPARC-V8 architecture. The resulting object code is in ELF format. This is the default choice of the `-xarch=` options.

`-xarch=v8a`

This option instructs the assembler to accept instructions defined in the SPARC-V8 architecture, less the `fsmuld` instruction. The resulting object code is in ELF format.

`-xarch=v8plus`

This option instructs the assembler to accept instructions defined in the SPARC-V9 architecture. The resulting object code is in ELF format, marked to indicate that it uses V9 instructions. It will not execute on a system with a V8 processor under a Solaris operating environment. It will execute on a system with a V9 processor under a Solaris operating environment. For more information regarding SPARC-V9 instructions, see Appendix E, “SPARC-V9 Instruction Set."

`-xarch=v8plusa`

This option instructs the assembler to accept instructions defined in the SPARC-V9 architecture, plus the instructions in the Visual Instruction Set (VIS). The resulting object code is in ELF format, marked to indicate that it uses V9/VIS instructions. It will not execute on a system with a V8 processor under a Solaris operating environment. It will execute on a system with a V9 processor under a Solaris operating environment. For more information about VIS instructions, see the "UltraSPARC Programmer’s Reference Manual" and the "UltraSPARC User’s Guide."

Disassembling Object Code

The `dis` program is the object code disassembler for ELF. It produces an assembly language listing of the object file. For detailed information about this function, see the man page `dis(1)`.

C