

# B

## Configuring the Driver

---

This appendix describes how to modify the driver configuration file. It contains the following sections:

§B.1: *Overview*

§B.2: *User-Modifiable Configuration Variables*

§B.3: *The Driver Configuration Language*

---

### B.1 Overview

You can change the default values of many compiler controls by modifying the configuration (`.cnf`) file. You can edit this file to assign values to the variables shown in §B.2: *User-Modifiable Configuration Variables*.

Directions for modifying the configuration file appear as comments in the file itself. Consult those comments and §B.3: *The Driver Configuration Language* for additional information about user-modifiable values.

---

**Note:** Before you modify the contents of the configuration file, we recommend you make a copy in a separate directory.

---

For additional information on using the driver configuration file, including default file names, see §2.5.5: *Using the Driver Configuration (.cnf) File*.

---

### B.2 User-Modifiable Configuration Variables

In the driver configuration file, the character `#` is a “comment” designator: any line that begins with `#` is a comment. You must remove comment designators to activate the following assignments. For example, to set a value for `HCDIR`, you must delete the `#` from the line that starts `#HCDIR=`.

---

**Caution:** If you modify the `.cnf` file beyond making the changes described in this section, you do so *at your own risk*. Generally, we do not support extensive customer modifications to the `.cnf` file. We are supporting your use of the High C/C++ compiler, not of the driver language.

---

The rest of this section documents some important variables in the `.cnf` file(s). This list is not exhaustive. See the `.cnf` file(s) provided with your distribution for additional variables and their default values.

### **ARGS — Specify defaults for command-line arguments**

Some operating systems place a limit on the length of the command line. You can preset a longer string of options by specifying them as parameters to `ARGS`, and the driver will process them as though you had specified them on the command line.

If there are options you always use, you can specify their values with `ARGS` so they take effect in all compilations.

Any toggle can be given a default value via `ARGS`. For example:

```
ARGS= -Hon=Pointers_compatible
```

Because these options are invisible on the actual command line, it is easy to forget about them. To display the options you have set, specify `-v` on the command line.

### **AS — Assembler variable**

For assembling. The driver assumes this variable is defined.

Within the definition, you specify the input source (`%s`) and the assembly output (`%o`). The driver provides the particular file names in the positions denoted by the `%` symbols.

### **CEXT — Specify C source file-name extension**

Default file-name extension for C source files (usually `.c`).

### **CPLUS — Specify C++ compilation**

Tells the compiler you intend to compile and link C++ files. When `CPLUS` is set to 1 (one), the High C++ header files and run-time library are included. `CPLUS` can also be set as an environment variable. See §8.1: *Compiling and Linking C and C++ Modules*.

**CPPEXT — Specify C++ source file-name extension**

Default file-name extension for C++ source files (usually `.cpp`, `.cc`, and `.c`). See command-line option `-Hcppext` in Chapter 3: *Using Compiler Options*.

**CPP — C preprocessor variable**

The definition of UNIX `cpp`. If this variable is defined, the driver invokes `cpp` as the preprocessor.

**HCDEBUG — Set compiler debug controls**

A variable that sets compiler debug controls when `-g` is specified on the command line. The driver passes the values assigned to `HCDEBUG` to the compiler. By default, `HCDEBUG` turns off toggle `Cross_jump`.

**HCDIR — Specify High C/C++ directory name**

Directory where you initiated the installation (can also be set as an environment variable).

**LINKER — Linker variable**

For linking. The driver assumes this variable is defined.

Within the definition, you specify the input objects (`%o`), the optional output load file (`%O`), the libraries, and the start-up file. The driver provides the particular file names in the positions denoted by the `%` symbols.

**TMPPREFIX — Specify alternate directory for temporary files**

On UNIX, temporary files required by the compiler are ordinarily created in a default directory, such as `/tmp`; on DOS, the temporary files are usually created in your current working directory. To designate an alternate directory for temporary files, set the `TMPPREFIX` variable in the `.cnf` file. For example:

```
TMPPREFIX= /cppfiles/my_temp
```

You can also set `TMPPREFIX` as an environment variable:

```
UNIX host    setenv TMPPREFIX /temp
```

```
DOS host    set TMPPREFIX=\temp
```

---

**Note:** On DOS, there must be no blanks in the `set` string beginning with `TMPPREFIX`.

---

**TOOLS DIR — Specify path to assembler and linker**

Directory where your assembler and linker reside (can also be set as an environment variable).

---

## B.3 The Driver Configuration Language

The driver configuration files are written in a “driver language” that allows a single driver program on any host processor to run any MetaWare compiler hosted on that processor.

**Caution:** If you modify the `.cnf` file beyond making the changes described in §B.2: *User-Modifiable Configuration Variables*, you do so *at your own risk*.

Generally, we do not support extensive customer modifications to the `.cnf` file. We are supporting your use of the High C/C++ compiler, not of the driver language.

---



---

### B.3.1 Driver Command Reference

This section describes some of the driver configuration language commands.

**\ — Continue to next line**

A backslash at the end of a line means the next line is a continuation of this line.

**XXX=x y z — Define variable as a string of tokens**

Defines variable `XXX` as the string of tokens `x y z`.

**\$YYY — Expand to the value of a defined variable**

Expands to the value of variable `YYY`, which must be defined with the `XXX=x y z` command described in the preceding example.

**\$?YYY — Expand to the value of a defined configuration-file variable**

Expands to the value of the variable `YYY`. `YYY` must be defined in the `.cnf` file.

**\$%eYYY — Expand to the value of a defined environment variable**

Expands to the value of variable *YYY*. *YYY* must be defined in the environment.

**\$-opt — Expand to the value of a command-line option -opt**

Expands to the value of *-opt* and any supplied argument to *-opt*.

Usually just the same as *-opt*, except that an error is reported if *-opt* was not given on the command line.

**\$?-opt — Expand to the value of -opt with optional arguments**

Expands to the value of the variable *-opt* followed by any argument supplied to *-opt*, if *-opt* was given on the command line. Otherwise it expands to nothing.

**\$-opt? — Expand to the value of argument to -opt**

Expands to the value of the argument supplied to *-opt* on the command line. If *-opt* was not given, an error is reported.

**\$?-opt? — Expand to the value of the argument to -opt**

Expands to the value of the argument supplied to *-opt* on the command line. If *-opt* was not given on the command line, this command expands to nothing.

**\$?:^ test ^ true\_part ^ false\_part ^ — Evaluate to true\_part if test is true; evaluate to false\_part if not**

Evaluates to the value of *true\_part* if *test* is true. If *test* is not true, evaluates to the value of *false\_part*.

The character “^” is arbitrary but must appear immediately after the “:” character. Whitespace within the token sequences is irrelevant.

**?-opt — Execute if -opt was given on the command line**

Execute the rest of the line if option *-opt* was given on the command line.

**?-opt? — Execute line if -opt given with an argument**

Execute the rest of the line if option *-opt* was given on the command line with a value, as in *-optXYZ*.

**?%eXXX — Execute line if xxx is defined**

Execute the rest of the line if *xxx* is defined in the environment.

**?<sup>=</sup> *tokens1* ^ *tokens2* ^ — Execute line if tokens are identical**

Execute the rest of the line if *tokens1* and *tokens2* are identical.

The character “^” is arbitrary but must appear immediately after the “=” character. Whitespace within the token sequences is irrelevant.

**?|<sup>^</sup> *test1* ^ *test2* ^ ... ^| — Alternate: true if any given test is true**

Alternation. True if and only if any of the *testi* are true.

The character “^” is arbitrary but must appear immediately after the “|” character. Whitespace within the token sequences is irrelevant.

**?!*test* — Execute line if *test* is not true**

Execute the rest of the line if *?test* does not hold; inverse of *?test*.

*test* is any of the tests described in preceding commands.

**#if *test1* ... #elif *test2* ... #else ... #endif — Conditional inclusion**

Conditional inclusion as in C. The tests are one or more of the *?test* items described in the preceding commands. For example,

```
#if ?-Hansi ?!$XYZ
```

includes text only if the **-Hansi** switch was not given on the command line and the XYZ variable is *not* defined.

**%; — Terminate a variable definition**

Terminates a variable definition. The rest of the line is processed as a new statement.

**#include *filename* — Include a file**

Includes *filename*. *filename* is taken relative to the directory of the .cnf file.

**#print *message* — Print a message to stderr**

Prints out a message to stderr.

**#exit *message* — Print a message to stderr and quit**

Prints out a message to stderr and then quits.

**# *xxx* — Start a comment to end-of-line**

Pound-sign (#) starts a comment to the end-of-line. This interpretation of # holds only when no previous interpretation (such as **#if**) holds.

**-opt — Define an option -opt**

Define option **-opt**. It can be interrogated later, as in **?-opt**, which evaluates to true if the option was given on the command line.

**-opt? — Define an option -opt with an argument**

Define option **-opt**. Such an option can have an argument immediately following it, as in **-Hsuffix=?** which allows **-Hsuffix=.o**, for example.

