

NAME

F – F compiler

USAGE

F [option] ... [*file*]...

DESCRIPTION

F is the Fortran Company / NAGWare F compiler. It translates programs written in F into executable programs, relocatable binary modules, assembler source files or C source files.

Note: *Some of these options are not available on all systems.*

The suffix of a filename determines the action **F** performs upon it. Files ending in **.f90** or **.f95** are taken to be F source files. Files ending in **.F90** or **.F95** are taken to be source form files requiring preprocessing by fpp. The file list may contain filenames of any form.

If a filename without a suffix is provided and there is no file of that name, F will look for a file with the suffix ".f95", and if that does not exist, the suffix ".f90".

Modules and include files are expected to exist in the current working directory or in a directory named by the **-I** option.

Options not recognised by F are passed to the load phase (gcc).

OPTIONS

- c** Compile only (produce .o file for each source file), do not link the .o files to produce an executable file.
- C** Compile code with all possible runtime checks. This option is a synonym for **-C=all**.
- C=check** Compile checking code according to the value of *check*, which must be one of:
 - all** (perform all checks),
 - array** (check array bounds),
 - calls** (check procedure references),
 - do** (check DO loops for zero step values),
 - none** (do no checking: this is the default),
 - present** (check OPTIONAL references), or
 - pointer** (check POINTER references).
- Dname** Defines *name* to fpp as a preprocessor variable. This is equivalent to passing the **-D** option directly to fpp, and affects only .F90 and .F95 files.
- dryrun** Show but do not execute commands constructed by the compiler driver.
- F** Preprocess all .F90 and .F95 files producing .f90 and .f95 output; no compilation is done.
- f77** Make external linkages compatible with the native f77 compiler where possible. On AIX, HP-UX and VMS, this suppresses the trailing underscore for those external names that do not clash with C language keywords or library functions.
- float-store** (Gnu C based systems only) Do not store floating-point variables in registers on machines with floating-point registers wider than 64 bits. This can avoid problems with excess precision.
- g** Produce information for interactive debugging by the host system debugger. Implies **-O0**.
- g90** Compile for debugging by *dbx90*, the Fortran-90 aware front-end to dbx. This produces a debug information (.g90) file for each Fortran source file. This option may be unavailable on some implementations.

- gc** Enables automatic garbage collection of the executable program.
- gline** Compile code to produce line number information in runtime error messages (note that this affects most, but not all, runtime error messages). This option increases both executable file size and execution time.
- hpf** Accept the extensions to Fortran specified by the High Performance Fortran Forum in HPF 1.0. These consist of the EXTRINSIC keyword, some additional intrinsic functions, and a large number of compiler directives. The compiler directives are checked for correctness but have no effect on compilation.
- I *pathname*** Add *pathname* to the list of directories which are to be searched for module information (.mod) files and INCLUDE files. The current working directory is always searched first, then any directories named in *-I* options, then the compiler's library directory (typically /usr/local/lib/F or /opt/F/lib).
- ieee=*mode*** Set the mode of IEEE arithmetic operation according to *mode*, which must be one of **full**, **nonstd** or **stop**.
 - ieee=full* enables all IEEE arithmetic facilities including non-stop arithmetic.
 - ieee=nonstd* disables IEEE gradual underflow, producing zero instead of a denormalised number; the resulting program may run faster. Non-stop arithmetic is also disabled, terminating execution on floating overflow, divide by zero or invalid operand.
 - ieee=stop* enables all IEEE arithmetic facilities except for non-stop arithmetic; execution will be terminated on floating overflow, divide by zero or invalid operand.

On DEC Alpha running Digital UNIX, the *-ieee* option must be specified when compiling all program units. On other machines and operating systems, the *-ieee* option must be specified when compiling the main program unit, and its effect is global. The default mode is *-ieee=stop*.
- info** Request output of information messages. The default is to suppress these messages.
- kind=*option*** Specify the kind numbering system to be used; *option* must be one of **byte** or **sequential**.

For *-kind=byte*, the kind numbers for INTEGER, REAL and LOGICAL will match the number of bytes of storage (e.g., default REAL is 4 and DOUBLE PRECISION is 8). Note that COMPLEX kind numbers are the same as its REAL components, and thus half of the total byte length in the entity.

For *kind=sequential* (the default), the kind numbers for all datatypes are numbered sequentially from 1, increasing with precision (e.g., default REAL is 1 and DOUBLE PRECISION is 2).

This option does not affect the interpretation of byte-length specifiers (an extension to Fortran 77).
- lx** Load with library libx.a. The loader will search for this library in the directories specified by *-Ldir* options followed by the normal system directories (see the ld(1) command).
- Ldir** Add *dir* to the list of directories for library files (see the ld(1) command).
- M** Produce module information files (.mod files) only.
- mdir *dir*** Write any module information (.mod) files to directory *dir* instead of the current working directory.
- o *output*** Name the output file *output* instead of a.out. This may also be used to specify the name of the output file produced under the *-c* and *-S* options.

- O** Normal optimisation, equivalent to *-O2*.
- ON** Set the optimisation level to *N*. The optimisation levels are:
 - O0** No optimisation. This is the default, and is the only optimisation level compatible with debugging.
 - O1** Minimal quick optimisation.
 - O2** Normal optimisation.
 - O3** Further optimisation.
 - O4** Maximal optimisation.
- Oassumed** This is a synonym for *-Oassumed=contig*.
- Oassumed=shape** Optimises assumed-shape array dummy arguments according to the value of *shape*, which must be one of
 - always_contig** Optimised for contiguous actual arguments. If the actual argument is not contiguous a runtime error will occur (the compiler is not standard-conforming under this option).
 - contig** Optimised for contiguous actual arguments; if the actual argument is not contiguous (i.e. it is an array section) a contiguous local copy is made. This may speed up array section accessing if a sufficiently large number of array element or array operations is performed (i.e. if the cost of making the local copy is less than the overhead of discontinuous array accesses), but usually makes such accesses slower. Note that this option does not affect dummy arguments with the TARGET attribute; these are always accessed via the dope vector.
 - section** Optimised for low-moderate accesses to array section (discontinuous) actual arguments. This is the default.

Note that CHARACTER arrays are not affected by these options.
- Oblock=N** Specify the dimension of the blocks used for evaluating the MATMUL intrinsic. The default value (only when *-O* is used) is 30, i.e. the arguments are processed in 30x30 blocks.
- Ounroll=N** Specify the depth to which simple loops and array operations should be unrolled. The default is no unrolling (i.e. a depth of 1) for *-O0* and *-O1*, and a depth of 2 for *-O* and higher optimisation levels. It can be advantageous to disable f95's loop unrolling if the C compiler normally does a very good job itself - this can be accomplished with *-Ounroll=1*.
- Ounsafe** Perform possibly unsafe optimisations that may depend on the numerical stability of the program.
- pg** Compile code to generate profiling information which is written at run-time to an implementation-dependent file (normally *gmon.out* or *mon.out*). An execution profile may then be generated using *gprof* (on Apollo, Sun and IBM Risc System), or *prof* (on DEC-stations and SGI). This option may be unavailable on some implementations.
- Qpath pathname** Change the f95 compiler library pathname from the default (typically */usr/local/lib/F* or */opt/F/lib*) to *pathname*.
- r8** Double the size of default REAL/COMPLEX, and on machines for which quadruple-precision REAL/COMPLEX are available, double the size of DOUBLE PRECISION. REAL/COMPLEX specified with explicit KIND numbers or byte lengths are unaffected - but since the KIND intrinsic returns the correct values, COMPLEX(KIND(0d0)) on a

machine with quad-precision floating-point will correctly select quad-precision COMPLEX.

This has no effect on INTEGER sizes, and so the compiler is not standard-conforming in this mode.

- s** Strip symbol table information from the executable file. This option is passed to ld so only has effect during the link step.
- S** Produce assembler (actually C source code). The resulting .c file should be compiled with the f95 command, not with the C compiler directly.
- save** This is equivalent to inserting the SAVE statement in all subprograms which are not declared RECURSIVE, thus causing all local variables in such subprograms to be statically allocated.

-target=*machine*

Specify the machine for which code should be generated and optimised. For Sun/SPARC, *machine* may be one of

V7	SPARCstation 1 et al.
V8	SPARCstation 2 et al.
super	SuperSPARC
ultra	UltraSPARC
native	the current machine

The default is to compile for SPARC V7. Note that programs compiled for later versions of the architecture may not run, or may run much more slowly, on an earlier machine. The *-target=super* and *-target=ultra* options do not work with Sun C prior to SC4.0. The *-target=native* option is not available with GCC.

For SGI/Irix 5, *machine* may be one of

mips1	R2000/R3000.
mips2	R4000 or later. Code thus produced will not run on R2000/R3000-based machines.

The default is to compile for R2000/R3000.

For Digital Alpha Unix, *machine* may be one of

ev4, ev5, ev56, pca56	the specified evolution of the Alpha chip
native	the current machine.

For HP9000/700, *machine* may be one of

native	the current machine
1.0, 1.1, 1.1a, 1.1b, 1.1c, 1.1d, 1.1e, 2.0	the specified revision of the PA-RISC architecture.

-tempdir *directory*

Set the directory used for temporary files to *directory*. The default is to use the directory named by the TMPDIR environment variable, or if that is not set, /tmp.

-time Report execution times for the various compilation phases.

-unsharedf95 Bind with the unshared (static) version of the f95 runtime library; this allows a dynamically linked executable to be run on systems where the NAGWare f95 compiler is not installed. This option is only effective if specified during the link step.

- v** Verbose. Print the name of each file as it is compiled.
- V** Print version information about the compiler.
- w** Suppress all warning messages. This option is a synonym for *-w=all*.
- w=class** Suppress the warning messages specified by *class*, which must be one of **all**, **obs**, **x77** or **x95**.
 - w=all* suppresses all warning messages.
 - w=obs* suppresses warning messages about the use of obsolescent features.
 - w=x77* suppresses extension warnings for common extensions to Fortran 77. These are TAB format, byte-length specifiers and Hollerith constants.
 - w=x95* suppresses extension warnings for extensions to Fortran 95 (currently just the HPF extensions).
- Woptions** The *-W* option can be used to specify the path to use for a compilation component or to pass an option directly to such a component. The possible combinations are:
 - W0=path** Specify the path used for the Fortran 95 front-end. Note that this does not affect the library directory; the *-Qpath* option should be used to specify that.
 - Wc=path** Specify the path to use for invoking the C compiler; this is used both for the final stage of compilation and for loading.
 - Wc,option** Pass *option* directly to the host C compiler when compiling (producing the *.o* file). Multiple options may be specified in a single *-Wc,* option by separating them with commas.
 - Wl,option** Pass *option* directly to the host C compiler when loading (producing the executable). Multiple options may be specified in a single *-Wl,* option by separating them with commas.
 - Wp=path** Specify the path to use for invoking the fpp preprocessor.
 - Wp,option** Pass *option* directly to fpp when preprocessing (only for *.F*, *.F90* and *.F95* files).
- xs** (Sun/SPARC option only) Store symbol tables for dbx in the executable (avoids the need to keep the object files for debugging).

FILES

a.out	Executable output file
<i>file.a</i>	Library of object files
<i>file.c</i>	C source file
<i>file.f90</i>	F source file
<i>file.f95</i>	F source file
<i>module-name.mod</i>	Compiled module information file
<i>file.o</i>	Object file
/opt/F/lib	NAGWare library directory on Sun Solaris (other than gcc) (may be changed by -Qpath); referred to as <i>library</i> hereafter.
/usr/local/lib/F	NAGWare library directory on other machines and operating systems.

<i>library/*.mod</i>	Compiled library module files (supplied by NAG)
<i>library/dope.h</i>	Dope vector header file
<i>library/f95.h</i>	Runtime library header file
<i>library/fcomp</i>	Fortran 95 compiler
<i>library/gc.o</i>	Garbage collector
<i>library/libf95.a</i>	Runtime library Note: On machines with the length of filenames restricted to less than 36 characters module information for some modules will be stored in files with different names. The supplied Technical Information note (TECHINFO) contains details if applicable.

DIAGNOSTICS

The diagnostics produced by F itself are intended to be self-explanatory. The loader, or more rarely the host C compiler, may produce occasional diagnostics.

Messages produced by the compiler are classified by severity level; these levels are:

Info	informational message, noting an aspect of the source code in which the user may be interested.
Warning	some questionable usage has been found in the user's code which may indicate a programming error.
Extension	some non-standard-conforming code has been detected but has successfully been compiled as an extension to the language. This has the same severity as "warning".
Error	the source code does not conform to the Fortran standard or does not make sense. Compilation continues after recovery.
Fatal	a serious error in the user's program from which the compiler cannot recover, the compilation is aborted immediately.
Panic	an internal inconsistency is found by one of the compiler's self-checks; this is a bug in the compiler itself and NAG should be notified.

COMPILER LIMITS

Maximum INCLUDE file nesting	= 20
Maximum number of INCLUDE file references per compilation	= 2047
Maximum DO loop nesting level	= 199
Maximum CASE construct nesting level	= 30
Maximum DATA-implied-DO loop nesting	= 20
Maximum array-structor-implied-DO loop nesting	= 20
Maximum number of dummy arguments	= 32767
Maximum number of arguments to MIN and MAX	= 20

I/O UNITS

Standard I/O Units

Error output	= 0
Input	= 5
Output	= 6

Default Record Lengths

Formatted	= 1024 characters
Unformatted	= 65536 bytes

AUTOMATIC FILE PRECONNECTION

All logical unit numbers are automatically preconnected to specific files. These files need not exist and will only be opened or created if they are accessed with READ or WRITE without an explicit OPEN. By default the specific filename for unit *n* is **fort.n**; however if the environment variable **FORT nn** exists its value is used as the filename. Note that there are two digits in this variable name, e.g. the variable controlling unit 1 is **FORT01** (unless the prefix has been changed, see the description of module **F90_PRECONN_IO**). However the default filename for unit 1, if no environment variable exists, is **"fort.1"**.

A file preconnected in this manner is opened with **ACCESS='SEQUENTIAL'**. If the initial READ or WRITE is an unformatted i/o statement, it is opened with **FORM='UNFORMATTED'** other it is opened with **FORM='FORMATTED'**. By default (see module **F90_PRECONN_IO**) a formatted connection is opened with **BLANK='NULL'** and **POSITION='REWIND'**.

Automatic preconnection applies only to the initial use of a logical unit; once CLOSED the unit will not be reconnected automatically but must be explicitly OPENed.

Note that this facility means that it is possible for a READ or WRITE statement with an **IOSTAT=** clause to receive an i/o error code associated with the implicit OPEN.

IEEE 754 ARITHMETIC SUPPORT

If no floating-point option is specified, any floating divide-by-zero, overflow or invalid operand exception will cause the execution of the program to be terminated (with an informative message and, on Unix, a core dump). Occurrence of floating underflow may be reported on normal termination of the program. On hardware supporting IEEE 754 standard arithmetic gradual underflow with denormalised numbers will be enabled. Note that this mode of operation is the only one available on hardware which does not support IEEE 754.

If the *-ieee=full* option is specified, non-stop arithmetic is enabled; thus REAL variables may take on the values +Infinity, -Infinity and NaN (Not-a-Number). If any of the floating exceptions listed above are detected by the hardware during execution, this fact will be reported on normal termination. The *-ieee=full* option must be specified when compiling the main program and has global effect; that is, it affects the entire executable program.

If the *-ieee=nonstd* option is specified, floating-point exceptions are handled in the default manner (i.e. execution is terminated). However, gradual underflow is **not** enabled, so results which would have produced a denormalised number produce zero instead. This option can only be used on hardware for which this mode of operation is faster. On most such machines it must be specified when compiling the main program and has global effect (like *-ieee=full*); however, on Digital UNIX (DEC Alpha) it only has full effect if used when compiling all program units - this is because on the Alpha architecture different (slower) code must be generated for arithmetic which is capable of producing denormalised numbers.

RANDOM_NUMBER

The random number generator supplied as the intrinsic subroutine RANDOM_NUMBER is the "good, minimal standard" generator described in "Random Number Generators: Good Ones Are Hard to Find" [CACM October 1988, Volume 31 Number 10, pp 1192-1201]. This is a parametric multiplicative linear congruential algorithm with the following parameters:

modulus: $2^{*}31-1$ (2147483647)
multiplier: 16807

This is a full-period generator. The seed is obtained from the time-of-day clock; viz $\text{time} * 16807 + 1$ where "time" is the number of seconds past midnight.

AUTOMATIC GARBAGE COLLECTION

The *-gc* option enables use of the runtime garbage collector. It is necessary to use this option during the link phase for it to have effect; specifying it additionally during the compilation phase can result in improved performance.

The supplied Technical Information note (TECHINFO) lists whether garbage collection is available for your system. If it is available, there will be a file "gc.o" in the compiler's library directory.

The collector used is based on version 4.11 of the publicly available general purpose garbage collecting storage allocator of Hans-J Boehm, Alan J Demers and Xerox Corporation, described in "Garbage Collection in an Uncooperative Environment" (H Boehm and M Weiser, Software Practice and Experience, September 1988, pp 807-820).

The copyright notice attached to their latest version is as follows:

Copyright 1988, 1989 Hans-J. Boehm, Alan J. Demers
Copyright (c) 1991-1996 by Xerox Corporation. All rights reserved.
Copyright (c) 1996 by Silicon Graphics. All rights reserved.

THIS MATERIAL IS PROVIDED AS IS, WITH ABSOLUTELY NO WARRANTY EXPRESSED OR IMPLIED. ANY USE IS AT YOUR OWN RISK.

Permission is hereby granted to use or copy this program for any purpose, provided the above notices are retained on all copies. Permission to modify the code and to distribute modified code is granted, provided the above notices are retained, and a notice that the code was modified is included with the above copyright notice.

Note that the "NO WARRANTY" disclaimer refers to the original copyright holders Boehm, Demers, Xerox Corporation and Silicon Graphics. The modified collector distributed in binary form with the F compiler is subject to the same warranty and conditions as the rest of the F compilation system.

The module **F90_GC** is provided; it contains functions and variables that can control the behaviour of the garbage collector.

DEBUGGING

For details on the Fortran-90-aware debugger see dbx90(1). This may not be available on some systems.

In general, host system debuggers, such as dbx, may be used successfully on Fortran code as the names of the original source files, plus line numbers, are passed through to the intermediate C files. In using such debuggers it should be noted that most local variables have an underscore appended to their names. It may be useful to look at the intermediate C code when debugging; this is produced by the -S option.

MODULES

To use a module it must be pre-compiled, or must be defined in the file prior to its use. When separately compiling a module the -c option should be specified.

Compiling a module creates a '.mod' file and usually a '.o' file. The '.mod' file is used by the compiler at compile time to provide information about module contents, the '.o' file (if generated) contains the code of any module procedures and must be specified when creating an executable file.

Note that the name of the '.mod' file will be the name of the module, the '.o' file will be named after the original source file.

When a precompiled module is USED the F compiler attempts to find its source file and, if that is successful, checks the modification times producing a warning message if the '.mod' file is out of date.

NAG-supplied modules are described in the man page nag_modules.

SEE ALSO

ar(1), cc(1V), f77(1V), f90_gc(3), f90_iostat(3), f90_kind(3), f90_preconn_io(3), f90_unix(3), f90_unix_dir(3), f90_unix_dirent(3), f90_unix_env(3), f90_unix_errno(3), f90_unix_file(3), f90_unix_proc(3), ieee_arithmetic(3), ieee_exceptions(3), ieee_features(3), ld(1), nag_modules(3).

BUGS

Please report any bugs found to "info@imagine1.com" along with any suggestions for improvements.

AUTHOR

Malcolm Cohen, The Numerical Algorithms Group Ltd., Oxford, United Kingdom. Hacked up by Walt Brainerd, The Fortran Company.