

NAME

pathf95, **pathf90** – Invoke the PathScale(TM) Fortran 77, 90, and 95 compilers

SYNOPSIS

pathf95 [-*option-list*] *files*

DESCRIPTION

This man page describes the **pathf95(1)** command, which invokes the PathScale(TM) Fortran 77, Fortran 90, and Fortran 95 compiler. The **pathf90(1)** command is an alias for **pathf95**. Typically, the **pathf95** command processes the input files named on the command line and generates a binary object file. The loader loads the binary object file and generates file **a.out**.

A complete list of available options can be found in the **eko(7)** man page.

The PathScale Fortran compiler recognizes code as being in either fixed-form (f77 72-column) format or free-form (f90) format, rather than Fortran 77, Fortran 90, or Fortran 95. By default, input files suffixed with **.f** or **.F** are assumed to be written in fixed source form (f77), and input files suffixed with **.f90**, **.F90**, **.f95**, or **.F95** are assumed to be written in free source form. You can use the **-freeform** or **-fixedform** flags to override these defaults. The suffixes with uppercase "F" cause the C preprocessor (**cpp**) to run, as if the **-cpp** option had been specified. Use the **-fcoco** option to run the ISO/IEC 1539-3 preprocessor, or the **-ftpp** option to run the traditional Fortran preprocessor. See the flags for more details.

Some **pathf95(1)** command options, for example, **-LIST**, **-LNO**, **-OPT**, **-TARG**, and **-TENV**, accept several suboptions and allow you to specify a setting for each argument. To specify multiple suboptions, either use colons to separate each suboption or specify multiple options on the command line. For example, the following command lines are equivalent:

```
pathf95 -LIST:notes=ON:options=OFF b.f
pathf95 -LIST:notes=ON -LIST:options=OFF b.f
```

Some suboptions to options of this type are specified with a setting that will either enable or disable the feature. To enable a feature, specify the suboption either alone or with **=1**, **=ON**, or **=TRUE**. To disable a feature, specify the suboption with either **=0**, **=OFF**, or **=FALSE**. For example, the following command lines are equivalent:

```
pathf95 -LNO:auto_dist:blocking=OFF:oinvar=FALSE a.f
pathf95 -LNO:auto_dist=1:blocking=0:oinvar=OFF a.f
```

For brevity, this man page shows only the **ON** or **OFF** settings to suboptions, but the compiler also accepts **0**, **1**, **TRUE**, and **FALSE** as settings.

The **pathf95** command accepts the following options:

- ###** Like -v, only nothing is run and args are quoted.
- ansi** Generate messages about constructs which violate standard Fortran syntax rules and constraints, plus messages about obsolescent and deleted features. This also disables all nonstandard intrinsic functions and subroutines, and implies **-ffortran2003**. Specifying **-ansi** in conjunction with **-fullwarn** causes all messages, regardless of level, to be generated.
- ar** Create an archive (instead of a shared object or executable) using **ar**.
- auto-use** *module_name*[,*module_name*] ...
Direct the compiler to behave as if a **USE** *module_name* statement were entered in your Fortran source code for each *module_name*. The **USE** statements are entered in every program unit and interface body in the source file being compiled (for example, **pathf95 -auto-use mpi_interface** or **pathf95 -auto-use shmем_interface**). Using this option can add compiler time in some situations.
- backslash** Treat a backslash as a normal character rather than as an escape character. When this option is used, the preprocessor will not be called.

-byteswapio

Swap bytes during I/O so that unformatted files on a little-endian processor are read and written in big-endian format (or vice versa.) In sequential unformatted files, this affects record headers as well as data. To be effective, the option must be used when compiling the Fortran main program. Setting the environment variable FILENV when running the program will override the compiled-in choice in favor of the choice established by the command **assign**(1).

-c

Disable the load step and write the binary object file to *file.o*.

-C

(Fortran) Perform runtime subscript range checking. Each subscript that is out of range generates a warning message. If you set the **F90_BOUNDS_CHECK_ABORT** environment variable to **YES**, the program aborts; otherwise, the program continues to run unless the out-of-range subscript itself causes a fatal error.

-CG

Option group to control code generation. See the **eko**(7) man page.

-colN

Specify the line width for fixed-format source lines. Specify **72**, **80**, or **120** for *N* (-col72, -col80, or -col120). By default, fixed-format lines are 72 characters wide. Specifying **-col120** implies **-extend-source** and recognizes lines up to 132 characters wide. For more information on specifying line length, see the **-extend-source** and **-noextend-source** options.

-convert *conversion*

Control the swapping of bytes during I/O so that unformatted files on a little-endian processor are read and written in big-endian format (or vice versa.) In sequential unformatted files, this affects record headers as well as data. To be effective, the option must be used when compiling the Fortran main program. Setting the environment variable FILENV when running the program will override the compiled-in choice in favor of the choice established by the command **assign**(1). Legal values of *conversion* are:

native

No conversion (the default)

big_endian

Files are big-endian

little_endian

Files are little-endian

-copyright

Show the copyright for the compiler being used.

-cpp

Preprocess all Fortran input source files with the C preprocessor **cpp**(1) before compiling. This preprocessor automatically expands macros outside of preprocessor statements.

The default is to run the C preprocessor if the input file ends in a **.F**, **.F90** or **.F95** suffix, but not to preprocess files ending in **.f**, **.f90**, or **.f95**.

For more information on controlling preprocessing, see the **-fcoco**, **-ftpp**, **-E**, **-macro-expand**, and **-nocpp** options.

-d-lines

Compile lines with a D in column 1.

-Dvar=[def][,var=[def] ...]

Define variables used for source preprocessing as if they had been defined by a **#define** directive. If no *def* is specified, **1** is used. For information on undefining variables, see the **-Uvar** option.

-default64

Set the sizes of default integer, real, logical, and double precision objects. This option is a synonym for the pair of options: **-r8 -i8**. Calling a routine in a specialized library, such as SCSL, requires that its 64-bit entry point be specified when 64-bit data are used. Similarly, its 32-bit entry point must be specified when 32-bit data are used.

-dumpversion

Show the version of the compiler being used and nothing else.

-E

Run only the source preprocessor files, without considering suffixes, and write the result to **stdout**. This option overrides the **-nocpp** option. The output file contains line directives. To generate an output file without line directives, see the **-P** option. For more information on controlling source preprocessing, see the **-fcoco**, **-cpp**, **-ftpp**, **-macro-expand**, and **-nocpp** options.

-extend-source

Specify a 132-character line length for fixed-format source lines. By default, fixed-format lines are 72 characters wide. For more information on controlling line length, see the **-coln** option.

-fb-create path

Used to specify that an instrumented executable program is to be generated. Such an executable is suitable for producing feedback data files with the specified prefix for use in feedback-directed compilation (FDO). The commonly used prefix is *<fbdata>*. This is **OFF** by default.

-fb-opt path

Specify the directory that contains the instrumentation output generated by compiling with **-fb-create** and then run your program with a training input set. Directs the compiler to use this information to better optimize the program. When the **-c** option is used to produce an object file that is eventually linked to form an instrumented executable, the **-fb-opt** option should also be specified.

-fcoco[=*setfile*]

Run the ISO/IEC 1539-3 conditional compilation preprocessor on input Fortran source files before compiling. This overrides the default whereby files suffixed with **.F**, **.F90**, or **.F95** are preprocessed with **cpp** but files suffixed with **.f**, **.f90** or **.f95** are not preprocessed.

If no *setfile* is specified, the preprocessor looks for *coco.set* in the current working directory. Any **-I** flags are passed to the preprocessor, and take precedence over the *setfile*.

Any **-D** flags are passed to the preprocessor to assign values to constants, overriding values assigned within the source files. If the flag contains "=", the value on the right side must be an integer, and the name on the left side must be declared as an integer constant within the source files. Otherwise, the name must be declared as a logical constant within the source files, and will be set true. Constants defined by **-D** should not be defined in the *setfile*.

-fdecorate path

Specify how to "decorate" external Fortran identifiers to generate linker symbols. Ordinarily we apply the rules established by options **-f[no-]underscoring** and **-f[no-]second-underscore**, but **-fdecorate** overrides those rules for specific identifiers. The file *path* should contain two blank- or tab-delimited tokens per line. The first token is a Fortran identifier and the second is the linker symbol to use for that identifier. An abbreviation is allowed in place of the second token: "0" says to append no underscore to the Fortran identifier, "1" says to append a single underscore, and "2" says to append two underscores if the Fortran identifier contains an underscore but otherwise to append one. If an identifier appears twice, the second rule overrides the first.

You may repeat this option to specify multiple files.

-ff2c-abi path

Use the GNU **f2c** ABI when calling any functions listed in the file at *path*. On the x86_64 platform, the **g77** compiler generates code that does not follow the documented platform ABI in some cases (involving functions returning complex or single-precision real values). You must use this flag if you are mixing code generated by **g77** with code generated by the PathScale Fortran compiler.

The format of an **f2c** ABI description file is simply a list of Fortran function names, one per line, without any of the trailing underscores that are added in object files. To generate files in this format, you can use the **fsymlist(1)** utility.

-ffortran2003

When you apply the Fortran intrinsic `real`, `double`, or `complex` to a `boz` constant such as `z'3ff00000'`, the compiler traditionally converts the constant to an integer and returns the real value whose magnitude matches that integer. This option makes each intrinsic behave as Fortran 2003 requires, returning the real value whose bit pattern matches the `boz` constant.

-ffortran-bounds-check

Check bounds.

-fixedform

Treat all input source files, regardless of suffix, as if they were written in fixed source form (f77 72-column format), instead of F90 free format. By default, only input files suffixed with `.f` or `.F` are assumed to be written in fixed source form.

-flist

Invoke all Fortran listing control options. The effect is the same as if all **-FLIST** options are enabled.

-FLIST: ...

Invoke the Fortran listing control group, which controls production of the compiler's internal program representation back into Fortran code, after IPA inlining and loop-nest transformations. This is used primarily as a diagnostic tool, and the generated Fortran code may not always compile. With the exception of **-FLIST:=OFF**, any use of this option implies **-flist**. The arguments to the **-FLIST** option are as follows:

Argument	Action
=<i>setting</i>	Enable or disable the listing. <i>setting</i> can be either ON or OFF . The default is OFF .

This option is enabled when any other **-FLIST** options are enabled, but it can also be used to enable a listing when no other options are enabled.

ansi_format=*setting*

Set ANSI format. *setting* can be either **ON** or **OFF**. When set to **ON**, the compiler uses a space (instead of tab) for indentation and a maximum of 72 characters per line. The default is **OFF**.

emit_pfetch=*setting*

Write prefetch information, as comments, in the transformed source file. *setting* can be either **ON** or **OFF**. The default is **OFF**.

In the listing, **PREFETCH** identifies a prefetch and includes the variable reference (with an offset in bytes), an indication of read/write, a stride for each dimension, and a number in the range from 1 (low) to 3 (high), which reflects the confidence in the prefetch analysis. `prefetch` identifies the reference(s) being prefetched by the **PREFETCH** descriptor. The comments occur after a read/write to a variable and note the identifier of the **PREFETCH-spec** for each level of the cache.

ftn_file=*file*

Write the program to *file*. By default, the program is written to *file.w2f.f*.

linelength=*n*

Set the maximum line length to *n* characters.

show=*setting*

Write the input and output filenames to **stderr**. *setting* can be either **ON** or **OFF**. The default is **ON**.

-f[no-]directives

-fno-directives ignores all directives (such as `!$OMP` or `C*$* PREFETCH_REF`) inside comments. The default is **-fdirectives**, which scans the comments for directives (although

certain directives may have no effect unless additional options, such as **-mp**, are present.)

-f[no-]math-errno

Do not set ERRNO after calling math functions that are executed with a single instruction, e.g. **sqrt**. A program that relies on IEEE exceptions for math error handling may want to use this flag for speed while maintaining IEEE arithmetic compatibility. This is implied by **-Ofast**. The default is **-fmath-errno**.

-f[no-]preprocessed

-fpreprocessed tells the preprocessor that input has already been preprocessed. Using **-fno-preprocessed** tells preprocessor that input has not already been preprocessed.

-f[no-]second-underscore

If **-funderscoring** is in effect, and the original Fortran external identifier contained an underscore, **-fsecond-underscore** appends a second underscore to the one added by **-funderscoring**. **-fno-second-underscore** does not append a second underscore. The default is both **-funderscoring** and **-fsecond-underscore**. See also **-fdecorate**.

-f[no-]underscoring

-funderscoring appends an underscore to each external Fortran identifier to generate a linker symbol. **-fno-underscoring** appends no underscores. The default is both **-funderscoring** and **-fsecond-underscore**. See also **-fdecorate**.

-f[no-]unsafe-math-optimizations

-funsafe-math-optimizations improves FP speed by violating ANSI and IEEE rules. **-fno-unsafe-math-optimizations** makes the compilation conform to ANSI and IEEE math rules at the expense of speed.

-fPIC

Generate position independent code, if possible. It is **OFF** by default.

-ffreeform

Treat all input source files, regardless of suffix, as if they were written in free source form. By default, only input files suffixed with **.f90**, **.F90**, **.f95**, or **.F95** are assumed to be written in free source form.

-ftest-coverage

Create data files for the **pathcov(1)** code-coverage utility. The data file names begin with the name of your source file:

SOURCENAME.bb

A mapping from basic blocks to line numbers, which **pathcov** uses to associate basic block execution counts with line numbers.

SOURCENAME.bbg

A list of all arcs in the program flow graph. This allows **pathcov** to reconstruct the program flow graph, so that it can compute all basic block and arc execution counts from the information in the **SOURCENAME.da** file.

Use **-ftest-coverage** with **-fprofile-arcs**; the latter option adds instrumentation to the program, which then writes execution counts to another data file:

SOURCENAME.da

Runtime arc execution counts, used in conjunction with the arc information in the file **SOURCENAME.bbg**.

Coverage data will map better to the source files if **-ftest-coverage** is used without optimization. See the gcc man pages for more information.

-ftpp

Run the traditional Fortran source preprocessor on input Fortran source files before compiling. This overrides the default whereby files suffixed with **.F**, **.F90**, or **.F95** are preprocessed with **cpp** but files suffixed with **.f**, **.f90** or **.f95** are not preprocessed.

This preprocessor does not expand macros outside of preprocessor statements unless you also specify **-macro-expand**.

- fullwarn** Request that the compiler generate comment-level messages. These messages are suppressed by default. Specifying this option can be useful during software development.
- g[N]** Specify debugging support and to indicate the level of information produced by the compiler. The supported values for **N** are:
- 0** No debugging information for symbolic debugging is produced. This is the default.
 - 1** Produces minimal information, enough for making backtraces in parts of the program that you don't plan to debug. This is also the flag to use if the user wants backtraces but does not want the overhead of full debug information. This flag also causes **--export-dynamic** to be passed to the linker.
 - 2** Produce additional debugging information for symbolic debugging. Specifying **-g** without a debug level is equivalent to specifying **-g2**. If there is no explicit optimization flag specified, the **-O0** optimization level is used in order to maintain the accuracy of the debugging information. If optimization options **-O1**, **-O2**, or **-O3** are explicitly specified, the optimizations are performed accordingly but the accuracy of the debugging cannot be guaranteed. If **-ipa** is specified along with option **-g2**, then IPA is disabled.
- GRA:** Option group to control global register allocation.
- help** List all available options. The compiler is not invoked.
- help:** Print list of possible options that contain a given string.
- in** Specify the length of default integer constants, default integer variables, and logical quantities. Specify one of the following:
- | Option | Action |
|---------------|--|
| -i4 | Specifies 32-bit (4 byte-) objects. Default. |
| -i8 | Specifies 64-bit (8 byte-) objects. |
- Idir** Specify a directory in which to search for "include" and "use" files. This is used for the following types of files:
- Files named in **INCLUDE** lines in the Fortran source file that do not begin with a slash (/) character
 - Files named in **#include** source preprocessing directives that do not begin with a slash (/) character
 - Files specified in **USE** statements
- The search for "include" files takes place in this order: first, in the same directory as the file containing the **INCLUDE** or **#include**; second, in the directories specified by **-Idir**; and third, in the standard directory, **/usr/include**.
- The search for "use" files takes place in this order: first, in the current working directory; second, in the directory specified by the **-module** option if that option was used; and third, in the directories specified by **-Idir**.
- ignore-suffix**
Compile all files as if they were Fortran source files. By default, the **pathf95(1)** command determines the type of processing necessary for an input file based on its suffix. Files that end in **.c**, for example, are compiled by **pathcc(1)**. When **-ignore-suffix** is specified, the compiler processes all files named as if they were all Fortran source files, regardless of suffix.
- [no-]intrinsic=name**
Add a procedure to (or remove a procedure from) the set of intrinsic functions and subroutines that the compiler recognizes. By default, the compiler recognizes only some of the intrinsics that it can support.
- The *name* can be the lower-case name of any intrinsic that the compiler can support, or it can

be an upper-case name representing a predefined "family" of intrinsics. You can use the options to "tune" the compiler to provide all the intrinsics a program needs, while eliminating the ones whose names conflict with those of the program's own functions and subroutines. The options may appear multiple times, and will be interpreted in order. For example, "-no-intrinsic=EVERY -intrinsic=G77 -no-intrinsic=abort" would remove all intrinsics, then add the family of G77 intrinsics, and then remove the individual intrinsic "abort".

Predefined families are:

EVERY

Every intrinsic that the pathf95 compiler can support

ANSI Intrinsics defined in the ANSI standard; this is the default for the **-ansi** option.

G77 Intrinsics known to the GNU compiler

PGI Intrinsics known to the PGI(TM) compiler

OMP Intrinsics defined by the OpenMP standard (automatically enabled by the **-mp** option; see the **eko(7)** man page for more information)

TRADITIONAL

Intrinsics known to pathf95 prior to version 2.0; this is the default in the absence of the **-ansi** option.

A family like "PGI" contains intrinsics supported by both pathf95 and the PGI compiler; that does not imply that pathf95 supports every intrinsic in the PGI compiler.

-ipa Invoke inter-procedural analysis (IPA). Specifying this option is identical to specifying **-IPA** or **-IPA:**. Default settings for the individual IPA suboptions are used.

-IPA[: ...] Control the application of inter-procedural analysis (IPA) and optimization. This includes inlining, common block array padding, constant propagation, dead function elimination, alias analysis, and other features. Specify **-IPA** with no arguments to invoke the inter-procedural analysis phase with default options.

If you have included IPA directives in your source code, the **-IPA** option must be specified in order for those directives to be honored.

If you compile and load in distinct steps, you must use at least **-IPA** for the compile step, and you must specify **-IPA** and the individual options in the group for the load step. For more information on the individual options in this group, see **ipa** in the **eko(7)** man page for more information.

-isystem dir

When the `cpp` preprocessor is in use, this option searches *dir* for header files, after all directories specified by **-I** but before the standard system directories. It also marks *dir* as a system directory, so that it gets the same special treatment as is applied to the standard system directories. With respect to Fortran itself, this option searches *dir* for modules, after all directories specified by **-I** but before any directories containing modules intrinsic to Fortran.

-keep Write all intermediate compilation files. *file.s* contains the generated assembly language code. *file.i* contains the preprocessed source code. These files are retained after compilation is finished. If IPA is in effect and you want to retain *file.s*, you must specify **-IPA:keeplight=OFF** in addition to **-keep**.

-keepdollar

Treat the dollar sign (\$) as a normal last character in symbol names.

-llibrary Search the library named **liblibrary.a** or **liblibrary.so**. The loader searches libraries in the order you specify.

-Ldirectory

Change the library search algorithm for the loader. For *directory*, specify the path to a directory that should be searched before using the default system libraries. You can specify multiple **-L** options on the command line. The library search algorithm searches these directories in left to right order.

-LANG: ...

Controls the language option group. The following sections describe the suboptions available in this group.

Argument	Action
-----------------	---------------

heap_allocation_threshold=size

Determine heap or stack allocation. If the size of an automatic array or compiler temporary exceeds *size* bytes it is allocated on the heap instead of the stack. If *size* is **-1**, objects are always put on the stack. If *size* is **0**, objects are always put on the heap.

The default is **-1** for maximum performance and for compatibility with previous releases.

IEEE_minus_zero=setting

Enable or disable the **SIGN(3I)** intrinsic function's ability to recognize negative floating-point zero (**-0.0**). Specify either **ON** or **OFF** for *setting*. The default is **OFF**, which suppresses the minus sign. The minus sign is suppressed by default to prevent problems from hardware instructions and optimizations that can return a **-0.0** result from a **0.0** value. To obtain a minus sign (-) when printing a negative floating-point zero (**-0.0**), use the **-z** option on the **assign(1)** command.

IEEE_save=setting

The Fortran standard requires that any procedure which accesses the standard IEEE intrinsic modules via a "use" statement must save the floating point flags, halting mode, and rounding mode on entry; must restore the halting mode and rounding mode on exit; and must OR the saved flags with the current flags on exit. Setting this option **OFF** may improve execution speed by skipping these steps.

recursive=setting

Invoke the language option control group to control recursion support. *setting* can be either **ON** or **OFF**. The default is **OFF**.

In either mode, the compiler supports a recursive, stack-based calling sequence. The difference lies in the optimization of statically allocated local variables, as described in the following paragraphs.

With **-LANG:recursive=ON**, the compiler assumes that a statically allocated local variable could be referenced or modified by a recursive procedure call. Therefore, such a variable must be stored into memory before making a call and reloaded afterwards.

With **-LANG:recursive=OFF**, the compiler can safely assume that a statically allocated local variable is not referenced or modified by a procedure call. This setting enables the compiler to optimize more aggressively.

-LNO: ... Specify options and transformations performed on loop nests by the Loop Nest Optimizer (LNO). The **-LNO** options are enabled only if **-O3** is also specified

on the **pathf95(1)** command line.

For information on the individual options in this group, see **eko(7)** man page. For information on the LNO options that are in effect during a compilation, use the **-LIST:all_options=ON** option.

- m32** Compile for 32-bit ABI, also known as x86 or IA32.
- m64** Compile for 64-bit ABI, also known as AMD64, x86_64, or IA32e.
- macro-expand**
Enable macro expansion throughout each file in source files preprocessed with the Fortran preprocessor. For this flag to have any effect, you must explicitly enable the use of the Fortran preprocessor using the **-ftpp** flag (the default preprocessor is the C preprocessor). Without this option specified, macro expansion is limited to preprocessor **#** directives in files processed by the Fortran preprocessor. When this option is specified, macro expansion occurs throughout the source file.
- march=<cpu-type>**
Compiler will optimize code for the selected cpu type: **opteron**, **athlon**, **athlon64**, **athlon64fx**, **barcelona**, **em64t**, **pentium4**, **xeon**, **core**, **anyx86**, **auto**. **auto** means to optimize for the platform that the compiler is running on, which the compiler determines by reading `/proc/cpuinfo`. **anyx86** means a generic x86 processor. Under 32-bit ABI, anyx86 is a processor without SSE2/SSE3/3DNow! support; under 64-bit ABI it is a processor with SSE2 but without SSE3/3DNow!. **Core** refers to the Intel Core Microarchitecture, used by 64-bit CPUs such as Woodcrest. The default is **auto**.
- mcmode=(small|medium)**
Select the code size model to use when generating offsets within object files. Most programs will work with **-mcmode=small** (using 32-bit data relocations), but some need **-mcmode=medium** (using 32-bit relocations for code and 64-bit relocations for data).
- mcpu=<cpu-type>**
Behaves like **-march**. See **-march**.
- mno-sse2**
This flag is only applicable to **-m32**. **-mno-sse2** is ignored under **-m64** with a warning.
- module dir**
Create the ".mod" file corresponding to a "module" statement in the directory *dir* instead of the current working directory. Also, when searching for modules named in "use" statements, examine the directory *dir* before the directories established by **-I***dir* options.
- msse2** Enable use of SSE2 instructions. This is the default under both **-m64** and **-m32**.
- msse3** Enable use of SSE3 instructions. Default is **ON** under **-march=barcelona**, **-march=em64t** and **-march=core**. Otherwise, it is **OFF** by default.
- msse4a** Enable use of SSE4A instructions. Default is **OFF**.
- mtune=<cpu-type>**
Behaves like **-march**. See **-march**.
- nocpp** Disable the source preprocessor.

See the **-fcoco**, **-cpp**, **-E**, and **-ftpp** options for more information on controlling pre-processing.
- nodefaultlibs**
Do no use standard system libraries when linking.

- noextend-source**
Restrict Fortran source code lines to columns 1 through 72.
See the **-col***n* and **-extend-source** options for more information on controlling line length.
- no-pathcc**
-no-pathcc turns off the `__PATHSCALE__` and other predefined preprocessor macros.
- nostartfiles**
Do not use standard system startup files when linking.
- nostdinc** Direct the system to skip the standard directory, `/usr/include`, when searching for `#include` files and files named on `INCLUDE` statements.
- nostdlib** No predefined libraries or startfiles.
- o out_file** Write the executable file to `out_file` rather than to `a.out`. By default, the executable output file is written to `a.out`.
- O[n]** Specify the basic level of optimization desired. *n* can be one of the following:
- 0** Turn off all optimizations.
 - 1** Turn on local optimizations that can be done quickly.
 - 2** Turn on extensive optimization. This is the default. The optimizations at this level are generally conservative, in the sense that they are virtually always beneficial, provide improvements commensurate to the compile time spent to achieve them, and avoid changes which affect such things as floating point accuracy.
 - 3** Turn on aggressive optimization. The optimizations at this level are distinguished from **-O2** by their aggressiveness, generally seeking highest-quality generated code even if it requires extensive compile time. They may include optimizations that are generally beneficial but may hurt performance.

This includes but is not limited to turning on the Loop Nest Optimizer, **-LNO:opt=1**, and setting **-OPT:ro=1:IEEE_arith=2:Olimit=9000:reorg_common=ON**
 - s** Specify that code size is to be given priority in tradeoffs with execution time.
If no value is specified for *n*, 2 is assumed.
- objectlist** Read the following file to get a list of files to be linked.
- Ofast** Equivalent to **-O3 -ipa -OPT:Ofast -fno-math-errno -ffast-math**. Use optimizations selected to maximize performance. Although the optimizations are generally safe, they may affect floating point accuracy due to rearrangement of computations. **-OPT:Ofast** effectively turns on the following optimizations: **-OPT:ro=2:Olimit=0:div_split=on:alias=typed**. **-OPT:Ofast** is also described in the `eko(7)` man page.

NOTE: **-Ofast** enables **-ipa** (inter-procedural analysis), which places limitations on how libraries and `.o` files are built. See the `eko(7)` man page for more information.
- OPT: ...** Controls miscellaneous optimizations. These options override defaults based on the main optimization level. For information on the individual options in this group, see the `eko(7)` man page.
- P** When used with **-E**, the source preprocessor will not generate `#` lines in the output.

-pad-char-literals

Blank pad all character literal constants that are shorter than the size of the default integer type and that are passed as actual arguments. The padding extends the length to the size of the default integer type.

-pathcc Define `__PATHCC__` and other macros.

-pg Generate extra code to profile information suitable for the analysis program **path-prof(1)**. You must use this option when compiling the source files you want data about, and you must also use it when linking. See the `gcc` man pages for more information.

-rreal_spec

Specify the default kind specification for real values.

Option	Kind value
---------------	-------------------

-r4	Use REAL(KIND=4) and COMPLEX(KIND=4) for real and complex variables, respectively. Default.
------------	---

-r8	Use REAL(KIND=8) and COMPLEX(KIND=8) for real and complex variables, respectively.
------------	--

-S Generate an assembly file, *file.s*, rather than an object file (*file.o*).

-shared DSO-shared PIC code.

-shared-libgcc

Force the use of the shared libgcc library.

-show Show phases as they are being invoked.

-show-defaults

Show the default options in the `compiler.defaults` file.

-showO Show what phases would be called, but don't invoke anything.

-showt Show time taken by each phase.

--static Same as **-static**.

-static Suppress dynamic linking at runtime for shared libraries; uses static linking instead.

-static-data

Statically allocate all local variables. Statically allocated local variables are initialized to zero and exist for the life of the program. This option can be useful when porting programs from older systems in which all variables are statically allocated.

When compiling with the **-static-data** option, global data is allocated as part of the compiled object (*file.o*) file. The total size of any *file.o* cannot exceed 2 GB, but the total size of a program loaded from multiple *.o* files can exceed 2 GB. An individual common block cannot exceed 2 GB, but you can declare multiple common blocks each having that size.

If a parallel loop in a multi-processed program calls an external routine, that external routine cannot be compiled with the **-static-data** option. You can mix static and multi-processed object files in the same executable, but a static routine cannot be called from within a parallel region.

-static-libgcc

Force the use of the static libgcc library.

-stdinc Predefined include search path list.

-subverbose

Produce diagnostic output about the subscription management for the compiler.

- TENV:** ... Specifies the target environment option group. These options control the target environment assumed and/or produced by the compiler. See **TENV** in the **eko(7)** man page for details about the options that are available.
- uvar** Make the default type of a variable undefined, rather than using default Fortran 90 rules.
- Uvar** Undefine a variable for the source preprocessor. See the **-Dvar** option for information on defining variables.
- v** Print (on standard error output) the commands executed to run the stages of compilation. Also print the version number of the compiler driver program and of the preprocessor and the compiler proper.
- version** Write compiler release version information to **stdout**. No input file needs to be specified when this option is used.
- w** Suppress warning messages.
- Wc,arg1[,arg2...]** Pass the argument(s) *argi* to the compiler pass *c* where *c* is one of [**pfibal**]. The *c* selects the compiler pass according to the following table:
- | Character | Name |
|-----------|--------------|
| p | preprocessor |
| f | front-end |
| i | inliner |
| b | backend |
| a | assembler |
| l | loader |
- Sets of these phase names can be used to select any combination of phases. For example, **-Wba,-o,foo** passes the option **-o foo** to the **b** and **a** phases.
- Wno-format-nonliteral** Do not warn if format string is not a string literal.
- Wno-larger-than-<number>** Do not warn if an object is larger than <number> bytes.
- woff** Turn off named warnings
- woffall** Turn off all warnings.
- woffnum** Specify message numbers to suppress. Examples:
- Specifying **-woff2026** suppresses message number 2026.
 - Specifying **-woff2026-2352** suppresses messages 2026 through 2352.
 - Specifying **-woff2026-2352,2400-2500** suppresses messages 2026 through 2352 and messages 2400 through 2500.
- In the message level indicator, the message numbers appear after the dash.
- woffoptions** Turn off warnings about options.
- Wsign-compare** Warn about signed/unsigned comparisons.

-Yc,path Set the *path* in which to find the associated phase, using the same phase names as given in the **-W** option. The following characters can also be specified:

- I** Specifies where to search for include files
- S** Specifies where to search for startup files (**crt*.o**)
- L** Specifies where to search for libraries

file.suffix[**90**] [*file.suffix*[**90**]] ...

File or files to be processed, where *suffix* is either an uppercase **F** or a lowercase **f** for source files. Files ending in **.i**, **.o**, and **.s** are also accepted. The Fortran source files are compiled, and an executable object file is produced.

The default name of the executable object file is **a.out**. For example, the following command line produces **a.out**:

```
pathf95 myprog.f
```

By default, several files are created during processing. The compiler adds a suffix to the *file* portion of the file name and places the files it creates into your working directory. See the FILES section for more information on files used and generated.

ENVIRONMENT VARIABLES

For information on environment variables, see the **eko(7)** man page.

FILES

The following is a file summary:

File	Type
a.out	Executable output file.
<i>file.a</i>	Object file archive.
<i>file.B</i>	Intermediate file written by the front-end of the compiler. To retain this file, specify the -keep option.
<i>file.f</i> or <i>file.F</i>	Input Fortran source file in fixed source form. If <i>file</i> ends in .F , the C preprocessor is invoked.
<i>file.f90</i> , <i>file.f95</i> , <i>file.F90</i> , or <i>file.F95</i>	Input Fortran source file in free source form. If <i>file</i> ends in .F90 or .F95 , the C preprocessor is invoked.
<i>file.i</i>	File generated by the source preprocessor. To retain this file, specify the -P option.
<i>file.mod</i>	Module file. Compiling a module generates both a module file, which must be available before compiling "use" statements that refer to that module, and an object file, which must be available when linking the program. When compiling multiple source files at once, you must order them so that each module is compiled before any "use" statement which refers to that module.
<i>file.o</i>	Object file.
<i>file.s</i>	Assembly language file. To retain this file, specify the -S option.
<i>file.so</i>	Dynamic Shared Object (DSO) library.

COPYRIGHT

Copyright (C) 2007, 2008 PathScale, LLC. All Rights Reserved.

Copyright (C) 2006, 2007 QLogic Corp. All Rights Reserved.

Copyright (C) 2003, 2004, 2005, 2006 PathScale, Inc. All Rights Reserved.

Copyright (C) 2000, 2001 Silicon Graphics, Inc. All Rights Reserved.

SEE ALSO

pathcc(1), **pathscale_intro(7)**, **eko(7)**, **explain(1)**, **assign(1)**, **fsymlist(1)**, **compiler.defaults(5)**, **pathopt2(1)**

PathScale Compiler Suite and Subscription Manager Install Guide

PathScale Compiler Suite User Guide

PathScale Compiler Suite Support Guide

PathScale Debugger User Guide

Online documentation available at <http://www.pathscale.com/docs.html>