NAME

opalcc -- Open PAL C wrapper compiler

SYNTAX

opalcc [-showme|-showme:compile|-showme:link] ...

OPTIONS

--showme
: This option comes in several different variants (see below). None of the variants invokes the underlying compiler; they all provide information on how the underlying compiler would have been invoked had *--showme* not been used. The basic *--showme* option outputs the command line that would be executed to compile the program. **NOTE:** If a non-filename argument is passed on the command line, the *-showme* option will *not* display any additional flags. For example, both "opalcc --showme" and "opalcc --showme my_source.c" will show all the wrapper-supplied flags. But "opalcc --showme -v" will only show the underlying compiler name and "-v".

--showme:compile
: Output the compiler flags that would have been supplied to the C compiler.

--showme:link
: Output the linker flags that would have been supplied to the C compiler.

--showme:command
: Outputs the underlying C compiler command (which may be one or more tokens).

--showme:incdirs
: Outputs a space-delimited (but otherwise undecorated) list of directories that the wrapper compiler would have provided to the underlying C compiler to indicate where relevant header files are located.

--showme:libdirs
: Outputs a space-delimited (but otherwise undecorated) list of directories that the wrapper compiler would have provided to the underlying linker to indicate where relevant libraries are located.

--showme:libs
: Outputs a space-delimited (but otherwise undecorated) list of library names that the wrapper compiler would have used to link an application. For example: "mpi open-rte open-pal util".

--showme:version
: Outputs the version number of Open MPI.

--showme:help
: Output a brief usage help message.

See the man page for your underlying C compiler for other options that can be passed through opalcc.

DESCRIPTION

Conceptually, the role of these commands is quite simple: transparently add relevant compiler and linker flags to the user's command line that are necessary to compile / link Open PAL programs, and then invoke the underlying compiler to actually perform the command.

As such, these commands are frequently referred to as "wrapper" compilers because they do not actually compile or link applications themselves; they only add in command line flags and invoke the back-end compiler.

Background

Open MPI is comprised of three software layers: OPAL (Open Portable Access Layer), ORTE (Open Run-Time Environment), and OMPI (Open MPI). There are wrapper compilers for each layer; each layer's wrapper only links in the libraries relevant for that layer. Specifically, each layer provides the following wrapper compilers:

OPAL
opalcc and *opalc++*

ORTE
ortecc and *ortec++*

OMPI
mpicc, *mpic++*, *mpicxx*, *mpiCC* (only on systems with case-senstive file systems), and *mpifort* (and its legacy/deprecated names *mpif77* and *mpif90*). Note that *mpic++*, *mpicxx*, and *mpiCC* all invoke the same underlying C++ compiler with the same options. All are provided as compatibility with other MPI implementations.

Fortran Notes

The Fortran wrapper compiler for MPI (*mpifort*, and its legacy/deprecated names *mpif77* and *mpif90*) can compile and link MPI applications that use any/all of the MPI Fortran bindings: *mpif.h*, the *mpi* module, and the *mpi_f08* module (assuming Open MPI was installed with support for each of these Fortran bindings). Specifically: it is no longer necessary to use different wrapper compilers for applications that use *mpif.h* vs. applications that use the *mpi* module -- just use *mpifort* for all Fortran MPI applications.

Note, however, that the Fortran compiler may require additional command-line options to enforce a specific Fortran dialect. For example, in some versions of the IBM XLF compiler, if xlf90 is the underlying Fortran compiler, *-qfixed* may be necessary to compile fixed-format Fortran source files.

Finally, note that *mpifort* will be inoperative and will return an error on use if Fortran support was not built into the MP Ilayer.

Overview

opalcc is a convenience wrappers for the underlying C compiler. Translation of an Open PAL program requires the linkage of the Open PAL-specific libraries which may not reside in one of the standard search directories of ld(1). It also often requires the inclusion of header files what may also not be found in a standard location.

opalcc passes its arguments to the underlying C compiler along with the -I, -L and -l options required by Open PAL programs.

The Open PAL Team *strongly* encourages using the wrapper compilers instead of attempting to link to the Open PAL libraries manually. This allows the specific implementation of Open PAL to change without forcing changes to linker directives in users' Makefiles. Indeed, the specific set of flags and libraries used by the wrapper compilers depends on how Open PAL was configured and built; the values can change between different installations of the same version of Open PAL.

Indeed, since the wrappers are simply thin shells on top of an underlying compiler, there are very, very few compelling reasons *not* to use *opalcc*. When it is not possible to use the wrappers directly, the *-showme:compile* and *-showme:link* options should be used to determine what flags the wrappers would have used. For example:

shell$ cc -c file1.c ‘mpicc -showme:compile‘

shell$ cc -c file2.c ‘mpicc -showme:compile‘

shell$ cc file1.o file2.o ‘mpicc -showme:link‘ -o my_mpi_program

NOTES

It is possible to make the wrapper compilers multi-lib aware. That is, the libraries and includes specified may differ based on the compiler flags specified (for example, with the GNU compilers on Linux, a different library path may be used if -m32 is seen versus -m64 being seen). This is not the default behavior in a standard build, but can be activated (for example, in a binary package providing both 32 and 64 bit support). More information can be found at:

https://github.com/open-mpi/ompi/wiki/compilerwrapper3264

FILES

The string that the wrapper compilers insert into the command line before invoking the underlying compiler are stored in a text file created by Open PAL and installed to *$pkgdata/opalcc-wrapper-data.txt*, where *$pkgdata* is typically *$prefix/share/openmpi*, and *$prefix* is the top installation directory of Open PAL.

It is rarely necessary to edit this file, but it can be examined to gain insight into what flags the wrappers are placing on the command line.

ENVIRONMENT VARIABLES

By default, the wrappers use the compilers that were selected when Open PAL was configured. These compilers were either found automatically by Open MPI's "configure" script, or were selected by the user in the CC, CXX, F77, and/or FC environment variables before "configure" was invoked. Additionally, other arguments specific to the compiler may have been selected by configure.

These values can be selectively overridden by either editing the text files containing this configuration information (see the **FILES** section), or by setting selected environment variables of the form "OPAL_value".

Valid value names are:

CPPFLAGS
: Flags added when invoking the preprocessor (C or C++)

LDFLAGS
: Flags added when invoking the linker (C, C++, or Fortran)

LIBS
: Libraries added when invoking the linker (C, C++, or Fortran)

CC
: C compiler

CFLAGS
: C compiler flags

CXX
: C++ compiler

CXXFLAGS
: C++ compiler flags

FC
: Fortran compiler

FCFLAGS
: Fortran compiler flags