ANL/MCS-TM-ANL-01/x Rev x

Installation and User's Guide to MPICH, a Portable Implementation of MPI Version 1.2.7 The ch_shmem device for Shared Memory Processors

by

Contents

Abstract

MPI (Message-Passing Interface) is a standard specification for message-passing libraries. MPICH is a portable implementation of the full MPI-1 specification for a wide variety of parallel and distributed computing environments. MPICH contains, along with the MPI library itself, a programming environment for working with MPI programs. The programming environment includes a portable startup mechanism, several profiling libraries for studying the performance of MPI programs, and an X interface to all of

  - Miscellaneous new `MPI_Info` and `MPI_Datatype` routines.

- `MPICH`

we developed for the NEC SX-4 [7]. The ch_shmem

instead.)

This will configure MPICH for the default device; this is usually the appropriate choice. Section 4.1 discusses the options that can be given to configure toto MPICH.

The output of configure is piped to tee; this program both writes the output to the file specified by its argument (here 'c.log') and tstandard output. If you have trouble with the configure or make step, the file 'c.log' will help identify any problems.

3.e MPICH:

```
% make |& tee make.log
```

This may take a while, depending on the load on your system and on your file server, it may take anywhere from a few minutes to an hour or more.

2.3 Running examples

1.Optional) Build and run a simple test program:

```
% cd examples/basic
% make cpi
% ../../bin/mpirun -np 4 cpi
```

At this point you have run an MPI program on your system. If you have trouble, see Section 6.

2.Optional) Put the distribution through its complete acceptance test (See Section 4.10 for how to do this).

3.Optional) If you wish to install MPICH

2.4 Sample MPI programs

The MPICH distribution contains a variety of sample programs, which are located in the MPICH source tree. Most of these will work with any MPI implementation, not just MPICH.

examples/baaic contains a few short programs in Fortran, C, and C++ for testing the simplest features of MPI.

examples/test contains multiple test directories for the various parts of MPI. Enter "make testing" in this directory to run our suite of function tests.

examples/perftest Performance benchmarking programs. See the script runmpptest for information on how to run the benchmarks. These are relatively sophisticated.

```
mpicc -o foo foo.o
mpif77 -o foo foo.o
mpicxx -o foo foo.o
mpif90 -o foo foo.o
```

-mpianim Build version that generates real-time animation.

```
setenv MPI_GLOBMEMSIZE 8388308
```

Large messages are transfered in fragments, so `MPI_GLOBMEMSIZE` does not limit the maxi-

```
MPE_Errors_call_dbx_in_xterm
MPE_Signals_call_debugger
```

These error handlers are located in the MPE directory. A configure option (-mpedbg) includes these error handlers into the regular MPI libraries, and allows the command-line argument -mpedbg to make MPE_Errors_call_dbx_in_xterm the default error handler (instead of MPI_ERRORS_ARE_FATAL).

3.5.3 Starting jobs with a debugger

The -dbg=<name of debugger> option to mpirun causes processes to be run under the

```
dbx a.out 1234
```

or

```
dbx -pid 1234 a.out
```

where 1234 is the process id[1].

To do this with gdb, start gdb and at its prompt do

```
file a.out
attach 1234
```

One can also attach the TotalView debugger to a running program (See Section 3.5.6 below).

Debugging with TotalView. You can set breakpoints by clicking in the left margin on a line number. Most of the TotalView GUI is self-explanatory. You select things with the left mouse button, bring up an action menu with the middle button, and "dive" into functions, variables, structures, processes, etc., with the right button. Pressing `cntl-?` in any TotalView window brings up help relevant to that window. In the initial TotalView window it brings up general help. The full documentation (The TotalView *User's Guide*) is available from the Etnus web site.

You switch from viewing one process to the next with the arrow buttons at the top-right corner of the main window, or by explicitly selecting (left button) a process in the root window to re-focus an existing window onto that process, or by diving (1 1 f 3 2.8845 (button))]TJ0 -13.55

There are several available log formats and `logviewer` selects the version of jumpshot appropriate for a particular logfile. See the MPE manual, distributed along with this manual, for more details.

3.7 Execution tracing

Execution tracing is easily accomplished using the `-mpitrace` command line argument while linking:

```
mpicc -c cpi.c
mpicc -o cpi -mpitrace cpi.o
```

3.8 Performance measurements

The ‘

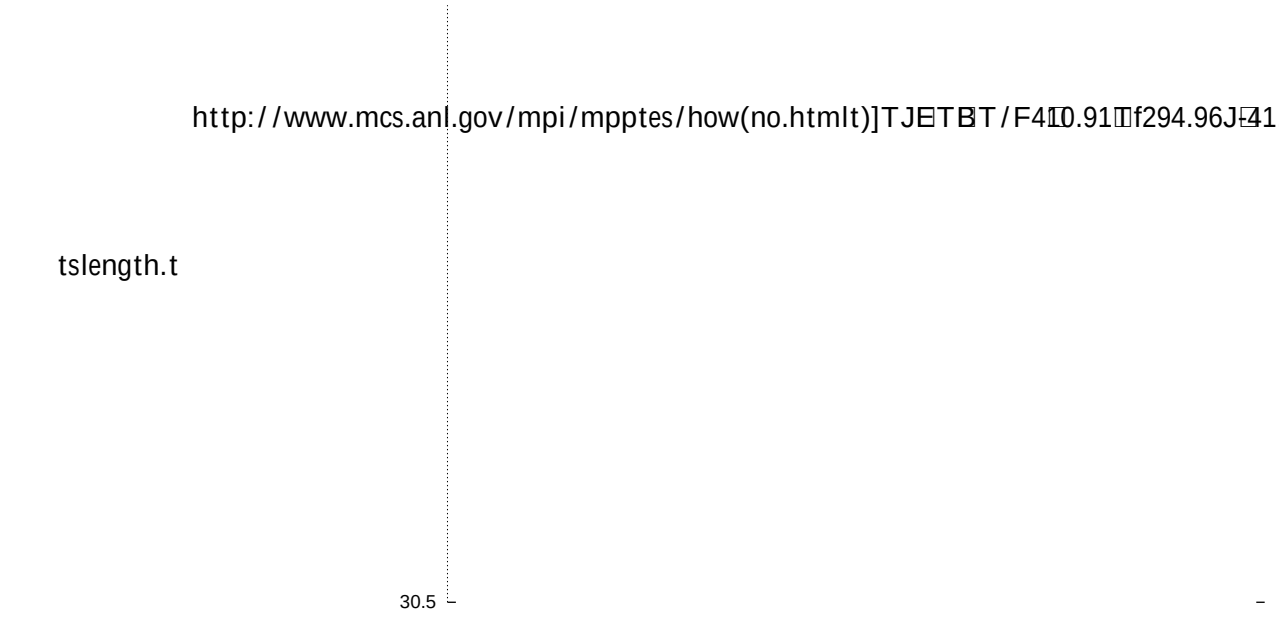

Figure 1: Sample output from `mpptest`

MPI implementation, not just `MPICH`. (See the `configure` file in the 'examples/perftest' directory.) More information is available at `http://www.mcs.anl.gov/mpi/mpptest`.

Benchmarking can be v.ry tricky. Soce common benchmarkingF410093tDdble
at

`--with-device=devname` Set the `MPICH` device to use. `devname` must be the name of one of the directories in the 'mpid' directory, such as `ch_p4`, `ch_shmem`, `globus2`, or `ch_p4mpd`.

`--enable-g` Add the `-g` option to the compile scripts. This is a prerequisite for using most debuggers, including `dbx`, and `gdb`.

`--enable-debug` Turn on support for the Totalview c Debugger. This allows Totalview to display information on message queues.

`--enable-sharedlib` Build both static and shared libraries for `MPICH`. This supports only a few systems, including those using `gcc` (e.g., most Linux Beowulf systems).

In addition, `configure`

be changed by the user.

Because Unix does not chain signals, there is the possibility that several packages will

```
setenv FC "f77 -n32 -mips4 -r10000"
configure --with-arch=IRIXN32 \
    -opt="-O2" \
    --with-device=ch_shmem

setenv CC "cc -64 -mips4 -r10000"
setenv FC "f77 -64 -mips4 -r10000"
configure --with-arch=IRIX64 \
    -opt="-O2" \
    --with-device=ch_shmem
```

```
setenv CC cc
configure --prefix=/usr/local/mpich-1.2.7
make
make install
cp util/mpichlib.conf /usr/local/mpich-1.2.7/etc/mpicc-cc.conf
setenv CC gcc
configure --prefix=/usr/local/mpich-1.2.7
make
cp util/mpichlib.conf /usr/local/mpich-1.2.7/etc/mpicc-gcc.conf
```

In this example, the default value of sysconfdir, `$prefix/etc`, is used.

For example, if MPICH was built with cc as the compiler but a user wanted to use gcc instead, either the commands

```
setenv MPICH_CC gcc
mpicc ...
```

or

```
mpicc -config=gcc ...
```

would case `mpicc` to load '`mpich-gcc.conf`' and use the appropriate definitions.

4.3 Fortran Compilers

MPICH provides support for both Fortran 77 and Fortran 90. Because MPICH is implemented in C, using MPICH from Fortran can sometimes require special options. This section discusses some of the issues. Note that `configure` tries to determine the options needed to support Fortran. You need the information in this section only if you have problems. Section 4.3.6 discusses how to support multiple Fortran compilers (e.g., `g77` and `pgf77`) with a single MPICH installation.

4.3.1 What if there is no Fortran compiler?

The `configure` program should discovu2ompileWhat there no Fortran compilers. You can

supp/foke‘

1. Change directory to ‘src/fortran’
2. Execute

```
setenv F77 pgf77
./configure --with-mpichconfig --with-flibname=mpich-pgf77
make clean
make
make install-alt
```

To use a particula2-13l/fortraoe-wiela2-13selirea2-13iea2-13ona2-13wiete

or

```
setenv MPICH_USE_SHLIB yes
mpicc -o cpi cpi.c
```

Using the environment variable MPICH_USE_SHLIB allows you to control whether shared libr F7 s are used without changing the compilation commands; this ca3Tf9sc ufulreor

```
-prefix=‘/usr/local/mpich-1.2.7/solaris/ch
```

The test codes in the ‘`mpich/examples/test`’ directory may be used with *any* implemen-

along with a ‘`Makefile.in`’. Other examples62(a)-there include a62(a)-simple parallel I/O program

– As a journal article in the Fall 1994 issue of the Journal of Supercomputing Applications [15] for MPI-1 and as a journal article in the International Journal

6.1 Things to try first

If something goes wrong, the first thing to do is to check the output of `configure` or `make` for some obvious problem, such as an improperly specified compiler or inadequate disk space.

Missing symbols when linking. The most common source of missing symbols is a failure of the MPICH configure step to determine how to pass command line arguments to

```
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/usr/local/mpich/lib/shared
mpirun -np 4 cpi
```

Unfortunately, this won't always work. Depending on the method that `mpirun` and `MPICH` use to start the processes, the environment variable may not be sent to the new process. This will cause the program to fail with a message like

6.6.2 Linux

1. Q: The configure step issues the message:

   ```
   checking that the compiler f77 runs... no
   Fortran compiler returned non-zero return code
   Output from test was
   f2ctmp_conftest.f:
      MAIN main:
   ```

   A:

to

```
case $cOPT in 2) $CC $G -o $OUTF $OFILES -lf2c -lm;; esac
rc=$?
trap 0
exit $rc
```

3. Q:

3. Q: When doing the link test, the link fails and does not seem to find any of the MPI routines:

A: We have been informed that there is a error in the `f77` script in some versions

2. Q: The ter2g□4Uea□TJ/F6.

```
sigset (code)
sighold (code)
*** Error code 1
```

A: You need to add the link option `-lV3`. The `ch_p4` device uses the System V signals on the HP; these are provided in the 'V3' library.

6.10.3 LINUX

1. Q: When linking a Fortran program, I get

```
Linking:
foo.o(.data+0x0): undefined reference to 'pmpi_wtime_'
```

A: This is a bug in the `pgf77` compiler (which is itself a workaround for a bug in the LINUX

If this works, then consider editing the value of LDFLAGS in the compiler scripts (e.g., mpicc) to include this option.

Unfortunately, each compiler has a different way of passing these arguments to the linker, and each linker has a different set of arguments for specifying the

A: This means that MPICH was built with the xlC compiler but that some of the machines ii7(h5(yc)27ours)-364‘t

4. Q: When running on an IBM SP, my job generates the message

```
Message number 0031-254 not found in Message Catalog.
```

   and then dies.

   A: If your user name is eight characters long, you may be experiencing a bug in the IBM POE environment. The only fix at the time this was written was to use an account whose user name was seven characters or less. Ask your IBM representative about PMR 4017X (poe with userids of length eight fails) and the associated APAR IX56566.

6.12 Programs fail at startup

6.12.1 General

1. Q:

(b) Use the secure server (`serv_p4`). See the discussion in the Users Guide.

(c) Redirect all standard output to a file. The MPE routine `MPE_IO_Stdout_to_file`

or it might compute

$$((a + b) + (c + d)) + ((e + f) + (g + h)),$$

where $a, b, \ldots$ are the values of

```
  Z()  { cerr << "*Z" << endl; }
  ~Z() { cerr << "+Z" << endl; }
};

Z z;

int main(int argc, char **argv) {
  MPI_Init(&argc, &argv);
  MPI_Finalize();
}
```

when running with the ch_p4

A: What is happening is that the TCP implementation on this platform is deciding that the connection has "failed" when it really hasn't. The current MPICH implementation assumes that the ion implementation will not close connections and has no code to reanimate failed connections. Future versions of MPICH will work around this problem.

In addition, some users have found that the single processor Linux kernel is more stable than the SMn kernel.

A: Your version of HP-UX limits the shell names to very short strings. Upshot is

enrich the MPICH implementation. More recent extensive contributions have been made by Omer Zaki (for Jumpshot). Anthony Chan has helped with SLOG and Jumpshot-3. David Ashton, who has developed the Windows NT version of MPICH and wat

A.1 Introduction

MPICH is a freely available, portable implementation of MPI, the Standard for message-passing libraries.

A.2 Installing MPICH

Building and installing MPICH often requires only

```
./configure --prefix=/home/me/mpich
make
make install
```

where the value of the `--prefix` argument to configure is the directory in which MPICH should be installed. See the *Installation Guide* for more detailed instructions.

A.3 Using MPICH

If this fails, then you may need a '`.rhosts`' or '`/etc/hosts.equiv`' file (you may need to see your system administrator) or you may need to use the p4 server. Another possible problem is the choice of the remote shell program; some systems have several. Check with your systems administrator about which version of rsh or remsh you should be using.

If your system allows a '`.rhosts`' file, do the following:

- Create a file .rhosts in your home directory
- Change the protection on it to user read/write only: chmod og-rwx .rhosts.
- Add one line to the .rhosts file for each processor that you want to use. The format is

```
host username
```

For example, if your username is doe and you want to user machines a.our.org and

Enabling rsh By default the rsh server is not installed, and it is necessary fe6

machine using the rsh or rlogin services. This packet filter, or firewall, is administered using

```
#       firewall; such entries will *not* be listed here.
:input ACCEPT
:forward ACCEPT
:output ACCEPT
-A input -s 0/0 67:68 -d 0/0 67:68 -p udp -i eth0 -j ACCEPT
-A input -s 0/0 67:68 -d 0/0 67:68 -p udp -i eth1 -j ACCEPT
-A input -s 0/0 -d 0/0 -i lo -j ACCEPT
#
525(67:68)-525(-p)-525(udp)-525(-i)-525(eth1)-525(-j)-525(ACCEPT)]T(25(-i)525(will)-ul□D[(:
#
513TD[(#)]b-525(-p)-525(udp)-525(-i)-525(eth1)-525(-j)-525(ACCEPT)]TJput
#
514TD[(#)]b-525(-p)-525(udp)-525(-i)-525(et51)-525(-j)-525(ACC4PT)]T(25(-i)EnTJ0□1ofJ0□1n25
#
0:1023TD[(#)]y-525(-p)-525(uREJECT(-j)-525(ACCEPT)]T5(-j)-525(ACCEPT)]TJput)-52tcpEPT
#
2049TD[(#)]y-525(-p)-525(uREJECT(-j)-525(ACCEPT)]T5(-j)-525(ACCEPT)]TJput)-52input #
0:1023TD[(#)] uREJECT(-j)-525(ACC4PT)]T5(-j)-525(ACCEPT)]TJput #
2049TD[(#)] uREJECT(-j)-525(ACCEPT)]T5(-j)-525(ACCEPT)]TJput
#
6000:6009TD[(#)]y-525(-p)-525(uREJECT(-j)-525(ACC4PT)]T5(-j)-525(ACCEPT)]TJput)-52tcpEPT
#
```

```
:output ACCEPT
```

Unfortunately for us, the default firewall configuration blocks some port ranges that our

or ssh), MPICH uses rsh to start the MPI processes. Depending on the particular Linux

A.9 SIGSEGV

A.15 Warning messages while building MPICH

Some compilers may generate a large number of warnings of the form

```
"commreq_free.c", line 70: warning #187: use of "=" where "==" may have been intended
```

There is nothing wrong with these statements. The compiler is warning about a legal, but often misused, feature of the C language. The statements have been crafted so that most compilers recognize that the "=" was used intentionally; unfortunately, some compilers insists on warning about this valid use of C and provide no way to indicate to the compiler that the warning is unnecessary.

A.16 MPMD (Multiple Program Multiple Data) Programs

MPICH, depending on the device, supports MPMD programs. However, the mpirund187:7.84.91Tfscrip4(

doc Assorted tools for producing documentation, together with this manual.

examples Directory containing further directories of example MPI programs. Of particular note are basic, with a few small examples to try first, test, with a test suite for exercising MPICH, and perftest

```
[-c++[=C++_COMPILER] ] [noc++]
[-opt=OPTFLAGS]
[-cc=C_COMPILER] [-fc=FORTRAN_COMPILER]
[-clinker=C_LINKER] [-flinker=FORTRAN_LINKER]
[-c++linker=CC_LINKER]
[-cflags=CFLAGS] [-fflags=FFLAGS] [-c++flags=CCFLAGS]
[-optcc=C_OPTFLAGS] [-optf77=F77_OPTFLAGS]
[-f90=F90_COMPILER] [-f90flags=F90_FLAGS]
[-f90inc=INCLUDE_DIRECTORY_SPEC_FORMAT_FOR_F90]
[-f90linker=F90_LINKER]
[-f90libpath=LIBRARY_PATH_SPEC_FORMAT_FOR_F90]
[-lib=LIBRARY] [-mpilibname=MPINAME]
[-mpe_opts=MPE_OPTS]
[-make=MAKEPGM ]
[-memdebug] [-ptrdebug] [-tracing] [-dlast]
[-listener_sig=SIGNAL_NAME]
[-cross]
[-adi_collective]
[-automountfix=AUTOMOUNTFIX]
[-noranlib] [-ar_nolocal]
[-rsh=RSHCOMMAND] [-D][-mpem11ı1.sTJ=owSTEM[-D][-mP4opts=MPE_OP-62.75e]
```

```
Packages that may be included with MPICH

--with-device=name     - Use the named device for communication.  Known
                         names include ch_p4, ch_mpl, ch_shmem, and globus2.
                         If not specified, a default is chosen.  Special
                         options for the device are specified after the
                         device name, separated by a colon.  E.g.,
                         --with-device=globus2:-flavor=mpi,nothreads
--with-romio[=OPTIONS] - Use ROMIO to provide MPI-I/O from MPI-2 (default).
                         The options include -file_system=FSTYPE, where
                         fstype can be any combination of nfs, ufs,
                         pfs (intel), piofs (IBM), hfs (HP), sfs (NEC), and
                         xfs (SGI), combined with '+'.  If romio is not
                         included, the Fortran 90 modules cannot be built.
--with-mpe             - Build the MPE environment (default)
--with-f90nag          - Choose the NAG f90 compiler for Fortran
                         (preliminary version intended for use *instead*
                         of a Fortran 77 compiler)
--with-f95nag          - Choose the NAG f95 compiler for Fortran
--with-cross=file      - Use the file for cross compilation. The file
                         should contain assignments of the form
                            CROSS_SIZEOF_INT=4
                         for each cross compilation variable.  The command
                            egrep 'CROSS_[A-Z_]*=' configure | sed 's/=.*//g'
                         will list each variable.

You can use --without-<featurename> to turn off a feature (except for
device).
Options for device ch_lfshmem:
--with-D[(Opt1at[:-an)5(-v]lfshmem:)]TJ0 11.  These-an-v[A-Z_]*appli[(Packageo)-525(of)-525(tl,)-525
```

```
Options for device ch_shmem:
--with-device=ch_shmem[:-usesysv]
The option '-usesysv' applies to the ch_shmem device, and causes the device
to attempt and use System V shared memory and semaphore routines, rather
than what would be chosen by default (often mmap or a system-specific method).

Options for device globus:
--with-device=globus[:-globusdir=GLOBUSDIR]
'-globusdir=GLOBUS' allows one to specify the location of an installed
version of Globus.  You can acquire Globus from http://www.globus.org ."

Options for device globus2:
*# GLOBUS_INSTALL_PATH must be set

Features that may be included with MPICH

--enable-c++          - Build C++ interfaces to the MPI-1 routines
                        (default)
```

that allows the user to produce variations of MPICH.

More notes on command-line parameters:

You can select a different C and Fortran compiler by using the '-cc' and 'fc' switches. The environment variables 'CC' and 'FC' can also provide values for these but their settings may be overridden by the configure script. Using '-cc=$CC -fc=$FC' will force configure to use those compilers.

The option '-opt' allows you to specify optimization options for the

(some AFS versions of 'rsh' have this bug), also give the option
'-rshnol'. These options are useful only when building a network version
of MPICH (e.g., '--with-device=ch_p4').

Special Tuning Options:

There are a number of options for tuning the behavoir of the ADI
(Abstract Device Interface) which is the low-level message-passir
interface. These should NOT be used unless you are sure you know what-]TJ0-11.955TD[(you)-525(are)-5

The option '-pkt_size=LENGTH' allows you to choose the message length at-]TJ0-11.96TD[(which)-525(th

```
        ch_shmem  (for shared memory systems, such as SMPs),
        ch_lfshmem(for shared memory systems, such as SMPs; uses
                   lock-free message buffers),
        ch_cenju3 (native NEC Cenju-3 calls)

The following devices were supported with ADI-1, but are currently
unsupported.  Please contact us if you are interested in helping us
support these devices:
        meiko(for Meiko CS2, using elan tport library), and
        nx        (for Intel Paragon),
        t3d          (for the Cray T3D, using Cray shmem library).
        ch_nc(native nCUBE calls, requires -arch=ncube),
        ch_cmmd   (native TMC CM-5 CMMD calls)
These are no longer distributed with the MPICH distribution.

Known architectures include (case is important)
        alpha(Compaq alpha)
        CRAY      (CRAY XMP, YMP, C90, J90, T90)
```

```
        included debugger scripts, and place it in the mpich/bin directory.
-ksq    Keep the send queue. This is useful if you expect later
        to attach totalview to the running (or deadlocked) job, and
        want to see the send queues. (Normally they are not maintained
        in a way which is visible to the debugger).
```

Options for the globus2 device:

With the exception of -h, these are the only mpirun options supported by the globus device.

```
-machinefile <machine-file name>
        Take the list of possible machines to run on from the
        file <machine-file name>
-np <np>
        specify the number of processors to run on
-dumprsl - display the RSL string that would have been
          used to submit the job.  using this option
          does not run the job.
-globusrsl <globus-rsl-file name>
        <globus-rsl-file name> must contain a Gvisible
```

When using the ch_p4 device, multiple architectures may be handled by giving multiple -arch and -np arguments. For example, to run a program on 2 sun4s and 3 rs6000s, with the local machine being a sun4, use

```
mpirun -arch sun4 -np 2 -arch rs6000 -np 3 program
```

This assumes that program will run on both architectures. If different executables are needed, the string '%a' will be replaced with the arch name. For example, if the programs are program.sun4 and program.rs6000, then the command is

```
mpirun -arch sun4 -np 2 -arch rs6000 -np 3 program.%a
```

If instead the executables are in different directories; for example, '/tmp/me/sun4' and '/tmp/me/rs6000', then the command is

```
mpirun -arch sun4 -np 2 -arch rs6000 -np 3 /tmp/me/%a/program
```

It is important to specify the architecture with -arch *before* specifying the number of processors. Also, the *first* arch

F.2 Obsolete Systems

Q:

To `configure` for the Intel Paragon, use

```
configure --with-device=ch_nx --with-arch=paragon
```

Two troubleshootings tip for the Paragon are

1. Q: I got the following messages when I tried to build on the Paragon:

   ```
   PGC-W-0115-Duplicate standard type (init.c: 576)
   PGC/Paragon Paragon Rel R5.0: compilation completed with warnings
   PGC-W-0115-Duplicate standard type (init.c: 576)
   PGC/Paragon Paragon Rel R5.0: compilation completed with warnings
   ```

   A: This is because the compiler doesn't handle `long long` but doesn't reject it either. It causes no harm.

2. Q: I get errors compiling or running Fortran programs.

   A:

   Fortran programs will need to use a *absolute* path for the '`mpif.h`' include file, due to a bug in the `if77` compiler (it susesstanda10.91 Tydir70(o334sin)-333(the)-333(P)27(awro3(or)-334(

```
mpireconfig Makefile
```

(`mpireconfig` is in the same directory as `mpirun`). This generates a new '`Makefile`' from '`Makefile.in`', with the correct parameters for the `MPICH` that was installed.

References

[14] William D. Gropp and Barry Smith. Chameleon parallel programming tools users manual. Technical Report ANL-93/23, Argonne National Laboratory, Argonne, IL, March 1993.