InterComm-2.0

Table of Contents

Intercomm 2.0

InterComm-2.0 uses MPI for inter-node communication via P^nMPI. PnMPI is distributed & automatically built with the InterComm-2.0 release.

OpenMPI >= 1.4.4 required

There is a bug with OpenMPI v 1.4.3 which interferes with InterComm:

"- Modified a memcpy() call in the openib btl connection setup to use
memmove() instead because of the possibility of an overlapping
copy (as identified by valgrind)."

1. Download
   a. via FTP (coming soon)

   ```
   cd $INSTALL_DIR/src
   wget http://cism.hao.ucar.edu/files/InterComm-2.0.tar.gz
   tar -zxvf InterComm-2.0.tar.gz
   ```

   b. via SVN (login required)

   ```
   svn checkout https://proxy.subversion.ucar.edu/cism_CSE/trunk/InterComm-2.0-wilsone
   ```

2. Configure, build and install

   ```
   ./configure CC=$CC CXX=$CXX FC=$FC MPICC=$MPICC MPICXX=$MPICXX MPIF90=$MPIF90 \
               --with-ppp=${INSTALL_DIR}/P++/install \
               --prefix=${INSTALL_DIR}/InterComm-2.0
   gmake
   gmake install
   ```

3. You are now ready to install the next prerequisite, Overture.

User Guide

Download the InterComm-2.0 user guide.

Troubleshooting

- When compiling InterComm, you get the error:

  ```
  ./configure: line 5466: syntax error near unexpected token `$f90_list'
  ./configure: line 5466: `  AC_PROG_F90($f90_list)'
  make: *** [config.status] Error 2
  ```

  Solution: Unpack a pristine copy of InterComm-2.0 and run `./bootstrap`. This will generate a new `configure` script which fixes the problem.

- When compiling, you get the error:

```
mpicc -c -g -O2 -g -DLinux -DINTEL -DDBGLEVEL=0x0000  -O3 -fPIC -DNO_FORT_PMPI_INIT -DNOSTATUS  services.
c
icc: command line warning #10120: overriding '-O2' with '-O3'
mpicxx core.o wrapper.o debug.o services.o -o libpnmpi.so  -shared -fPIC -lc -O3 -ldl
ld: /act/mvapich2/intel/lib/libmpich.a(attr_delete.o): relocation R_X86_64_32 against `MPIR_ThreadInfo'
can not be used when making a shared object; recompile with -fPIC
/act/mvapich2/intel/lib/libmpich.a: could not read symbols: Bad value
gmake[4]: *** [libpnmpi.so] Error 1
gmake[4]: Leaving directory `/drbd/home/pschmitt/opt-mvapich2-intel-11.1/src/InterComm-2.0/src/pnmpi/src'
gmake[3]: *** [src] Error 2
gmake[3]: Leaving directory `/drbd/home/pschmitt/opt-mvapich2-intel-11.1/src/InterComm-2.0/src/pnmpi'
gmake[2]: *** [all-recursive] Error 1
gmake[2]: Leaving directory `/drbd/home/pschmitt/opt-mvapich2-intel-11.1/src/InterComm-2.0/src'
gmake[1]: *** [all] Error 2
gmake[1]: Leaving directory `/drbd/home/pschmitt/opt-mvapich2-intel-11.1/src/InterComm-2.0/src'
gmake: *** [all-recursive] Error 1
```

Solution: edit `InterComm-2.0/src/pnmpi/src/Makefile`; find/replace "`$(MPICXX)`" with "`ld`"; Re-run `make`. (source)

- When running an InterComm-2.0 application, you get the error:

```
WARNING: This Pcontrol option is not supported (enable EXPERIMENTAL_UNWIND)

PnMPI Error: Cannot load virtual module
IC_Initialize failed at rank : 0
IC_Initialize with...
  XJD : hello.xjd
  Prog: serialF90Program
serial.C: IC_Recv_local('parallelSize')
WARNING: This Pcontrol option is not supported (enable EXPERIMENTAL_UNWIND)

PnMPI Error: Cannot load virtual module
serial.C: IC_Initialize status: -1
serial: xjd_handle.c:1831: IC_Recv_local: Assertion `xjd!=((void *)0)' failed.

=====================================================================================
=   BAD TERMINATION OF ONE OF YOUR APPLICATION PROCESSES
=   EXIT CODE: 134
=   CLEANING UP REMAINING PROCESSES
=   YOU CAN IGNORE THE BELOW CLEANUP MESSAGES
=====================================================================================
APPLICATION TERMINATED WITH THE EXIT STRING: Aborted (signal 6)
```

  - Solution: tbd. . . Possibly install the "unwind" library and set `EXPERIMENTAL_UNWIND=enabled` in the PnMPI build system (src/pnmpi/Makefile)?

Debugging InterComm-2.0

InterComm-2.0 was developed at the University of Maryland by the Chaos group. The code is unsupported and debugging InterComm can be full of chaos. The library works very well on some systems (typically using the Intel-12.x and OpenMPI-1.4.4). However, the code can fail in a variety of confusing ways. If you happen to see an error message, it might look like

```
IC_Setprogname: dlopen error
ERROR: NULL IC_getSubComm pointer from virtual
IC_Initialize failed at rank : 0
```

If InterComm still fails after trying all the obvious things (ie. try `source $INTERCOMM/lib/build.env` to set proper environment variables), then you may need to get your hands dirty. The first step is to build InterComm-2.0 with debug messages enabled. Extract a fresh copy of InterComm-2.0 source code and pass along an additional flag to configure: the `--enable-debug` flag.

Debugging PnMPI

When `--enable-debug` doesn't give you enough information, you can try enabling PnMPI debug messages. Try setting DEBUGLEVEL in `src/pnmpi/common/Makefile.common`. Sadly, there is no documentation on this. However, looking at the PnMPI source for debug.h sheds some light. There are several levels of verbosity you can set in a hexadecimal system:

```
#define DBGLEVEL1  0x0001  /* entry and exit prints */
#define DBGLEVEL2  0x0002  /* module load and instantiation */
#define DBGLEVEL3  0x0004  /* entry and exit of layers */
#define DBGLEVEL4  0x0008  /* arguments and parse information */
```

For very verbose messages, sum the DBGLEVEL hex numbers. The max verbosity would have `DEBUGLEVEL=0x000F` (1+2+4+8) in `src/pnmpi/common/Makefile.common`.

Testing PnMPI

There is a sample program distributed with PnMPI. Look in `src/pnmpi/demo`. Use `gmake` to build executables. Run programs with

- MPI:
  - mpirun -np 2 ./simple
  - mpirun -np 1 ./simple : -np 1 ./fdemo
- PnMPI:
  - mpirun -np 2 ./simple-pn
  - mpirun -np 1 ./simple : -np 1 ./fdemo-pn

This can be useful to determine where there's a problem with InterComm or its PnMPI dependency.