instructions

todo

- unzip source code packages and install.
- develop and test the code locally.
- deploy code on stampede2.

MPICH links

- references
- main page
- main page
- installation
- downloads
- tutorials
- developer documentation

TACC and stampede2 links

- user guide
- system access
- workshops

MPICH on Local Machine

This section includes all the material documented so far to run MPI application on a local machine.

Installation From Source

```
cd mpich-4.0.2/
./configure --prefix=/home/smerx/git/cse5351/hw/hw05/mpich-install/ --
disable-fortran --disable-cxx
make; sudo make install
export PATH=/home/smerx/git/cse5351/hw/hw05/mpich-install:$PATH
mpiexec --version
```

Installation Via App Manager

Install MPICH on linux via Debian package manager.

```
sudo apt install mpich
```

compile and run

Compile simple source code (no linking needed) and execute on local machine.

```
cd ./src
mpicc main.c -o ../out/hello_world
mpirun -np 5 ../out/hello-world
```

On TACC

Login

Log in to Stampede2 using SSH and use **** as password:

```
ssh mojra@stampede2.tacc.utexas.edu
```

Load Module and Environment Setup

Load the MPICH2 module:

```
module help swr     # show help text for software package swr
module help         # show help text for the module system itself
module load intel/19.1.1 gcc/9.1.0 # load required compilers
module load mvapich2/2.3.7 # load MPI module
module save         # save module config for later 'restore' and use.
module restore      # load personal default module
cd $SCRATCH         # use this directory as main workspace
pwd         # check and confirm workspace address
mkdir test && cd test
touch test.c        # c
vim test.c
```

File Transfer

Transfer the source code file using 'rsync'.

```
localhost$ rsync       mybigfile
bjones@stampede2.tacc.utexas.edu:\$WORK/data
localhost$ rsync -avtr mybigdir
bjones@stampede2.tacc.utexas.edu:\$WORK/data
```

Build Source Code

Compile and link (build) the source code.

```
mpicc test.c -o test  # C source, full build
mpicc -show  # Show compile line generated by call to mpicc; similarly for
other wrappers
```

or alternatively, the HEADER file and precompiled INC directory needs to be included in compile and linking commands. This method uses "mpich" library instead of using "mpicc" compiler application.

```
icc        -c main.c -I${WORK}/mylib/inc -I${TACC_HDF5_INC}
# compile
icc main.o -o myexe  -L${WORK}/mylib/lib -L${TACC_HDF5_LIB} -lmylib -lhdf5
# link
```

Launch the MPI Application

Use the following flags tih

Option	Argument	Description
-p	Title	Queue name.
-J	job_name	Job name.
-N	total_nodes	Total number of nodes to use (required).
-n	total_tasks	Total number of tasks (required).
-o	output_file	Output file.
-e	error_file	Standard error file.
-t	hh:mm:ss	Wall clock time for job (required).
--mail-user=	email_address	Specify the email address to use for notifications.

Use SBATCH to assign nodes and tasks and use 'ibrun' to execute binary.

```
#SBATCH -N 4 # num cores (tasks)
#SBATCH -n 1 # num nodes
ibrun test
```

To open an interactive environment for command-line debugging use 'idev'.

```
idev -N 4 -n 1 # run on 4 nodes (-N), 1 tasks each (-n)
ibrun test -o test.out -e test.err
```

Environment Setup for MPICH Dev and Debugging

```
ssh -X mojra@stampede2.tacc.utexas.edu # use -X for DDT GUI so X11 forwards
msgs
cd $SCRATCH && cd test
module load ddt_skx intel/19.1.1 gcc/9.1.0  mvapich2/2.3.7 # load required
compilers
mpicc  -g -O0 test.c -o test # compile with debug and zero optimization
options
idev -N 4 -n 1 # run on 4 nodes (-N), 1 tasks each (-n)
ddt test


# on window 01
ssh -X mojra@stampede2.tacc.utexas.edu # nbug
ssh mojra@stampede2.tacc.utexas.edu
cd $SCRATCH
cd test
pwd && ls
module load ddt_skx intel/19.1.1 gcc/9.1.0  mvapich2/2.3.7 # --- nbug
module load intel/19.1.1 gcc/9.1.0  mvapich2/2.3.7
mpicc  -g -O0 test.c -o test # nbug
mpicc  test.c -o test

# on second 02
ssh -X mojra@stampede2.tacc.utexas.edu # nbug
ssh mojra@stampede2.tacc.utexas.edu
cd $SCRATCH
cd test
pwd && ls
idev -N 1 -n 2 -t 0:15:0
ddt_skx test # nbug
ibrun test
```

Using idev Development and DDT Debugging Apps