Compiling, Running and Debugging NAME

Prerequisites

Compulsory

- Fortran compiler - If using ifort it will need to be v17 or higher New for v8.0
- C compiler - Linux distributions come with a C compiler.
- An MPI library and launcher - Not sure if you have an MPI library and launcher? Try running the simple MPI_Hello_World. If you need them see install Libraries New for v8.0

Table of Contents

Optional

See install Libraries

- ecCodes New for v8.1
- NetCDF New for v8.1

Red Hat Enterprise Linux 7 (RHEL 7) i.e Desktop

Compiling interactively

Set the environment. New for v8.0

```
>module list         # If ifort/17.0_64  is not listed. Then
load ifort v17:  module load ifort/17.0_64
>export MPICH_DIR=/home/h03/apdg/lib/mpich/3.2.1/ifort-17.0.1-
gcc-4.8.5
>export PATH=$MPICH_DIR/bin:$PATH
>export LD_LIBRARY_PATH=$MPICH_DIR/lib:$LD_LIBRARY_PATH
```

Check out the code, "Compile and run the Preprocessor" then Compile NAME

```
export src=~/NAME_trunk         # Replace with where you would
like your NAME build
>fcm checkout https://code.metoffice.gov.uk/svn/name/main/trunk
$src
>cd $src/Code_Preprocessor
>./LinuxIntelRelease 2>&1 | tee ~/Preprocessor.log
>cd $src/Code_NameIII
>make 2>&1 | tee ~/make.log
....
```

Running Interactively

Get the environment ready for running NAME

```
>cd $src/Executables_Linux
>export OMP_STACKSIZE=32m
>ulimit -s unlimited
```

Run NAME through the launcher. New for v8.0

```
>mpiexec -n 1 ./nameiii_64bit_par.exe name_input_file.txt
```

Batch running with SLURM i.e SPICE

Create a script called `run_name_spice.sh` containing:

```
#!/bin/bash -l
#SBATCH --mem=40000
#SBATCH --ntasks=12
#SBATCH --output=/path/to/directory/out.txt
#SBATCH --time=360
#SBATCH --export=NONE

ulimit -s unlimited
export OMP_STACKSIZE=32m
export MPICH_DIR=/home/h03/apdg/lib/mpich/3.2.1/ifort-17.0.1-
gcc-4.8.5
export PATH=$MPICH_DIR/bin:$PATH
export LD_LIBRARY_PATH=$MPICH_DIR/lib:$LD_LIBRARY_PATH

mpiexec -n 1 ./nameiii_64bit_par.exe maininput.txt
```

Submit the script.

```
>sbatch run_name_spice.sh
>sview     # Monitor the progress of your job.
>squeue    # To examine the queue.
>sacct     # To view the actual versus allocated resource.
```

Debugging

The old debugging pages can be seen at: wiki:NAMEDebugging

Write statements will need the buffers flushed. For example:

```
  write(6,*) 'nY:       ', nY
  write(6,*) 'HGrid%nY: ', HGrid%nY
  flush(6)
```

Compile the debug version of NAME

```
>cd $src/Code_NameIII
>make COMPILERMODE=Debug
```

Get the environment ready for running NAME

```
>cd $src/Executables_Linux
>export OMP_STACKSIZE=32m
>ulimit -s unlimited
```

Run the debug version of NAME through the launcher. New for v8.0

```
>mpiexec -n 1 ./nameiiiDebug_64bit_par.exe name_input_file.txt
```

Using GDB

If the diagnostic information is not indicative of the problem, then launch NAME with the debugger gdb. New for v8.0

```
>cd $src/Executables_Linux
>mpiexec -n 2 xterm -e gdb ./nameiiiDebug_64bit_par.exe
```

Run NAME in gdb

```
(gdb) run name_input_file.txt
```

To get a back trace

```
(gdb) bt
```

Cray Linux Environment (CLE) ie. XCE, XCF, XCS and Archer

Batch Compile with PBS

Check out the code from the mirror repository (You cannot connect directory to the repository for IT security reasons).

```
>export src=~/NAME_trunk_mirror    # Replace with where you
would like your NAME build
>fcm checkout svn://fcm1/name.xm_svn/main/trunk $src
>cd $src
```

Create a file called `compile_name.sh` in the `$src/Code_NameIII` directory containing:

```
#!/bin/bash --login

#PBS -l select=1:ncpus=1:mem=2GB
#PBS -l walltime=00:05:00
#PBS -j oe
#PBS -V

module swap PrgEnv-cray PrgEnv-intel
module swap intel intel/17.0.7.259

set -xu

cd $src/Code_Preprocessor
. CLEIntelRelease
```

```
cd $src/Code_NameIII

make clean
make COMPILEROPTIONS=IntelCLE
make clean
```

Then submit the job setting the source directory and the queue.

```
>cd $src/Code_NameIII
>chmod u+x compile_name.sh
>queue=shared                # For Archer this is queue=ppn
>qsub -q $queue -o compile_name.log compile_name.sh
```

Batch running with PBS

Create a script called `run_name.sh` in the `$src/Runs` directory to run NAME.

```
#!/bin/bash --login

#PBS -l select=1:ncpus=16:mem=32GB
#PBS -l walltime=00:30:00
#PBS -j oe
#PBS -V

module swap PrgEnv-cray PrgEnv-intel
module swap intel intel/17.0.7.259

cd $src/Executables_Linux
export OMP_STACKSIZE=32m
export KMP_AFFINITY=disabled          # Switching off the intel
affinity as it can cause problems with aprun which also handles
affinity.
export OMP_NUM_THREADS=16
export MPI_TASKS=1

aprun -n $MPI_TASKS -cc none -d $OMP_NUM_THREADS
./nameiii_64bit_par.exe $src/Runs/name_input_file.txt
```

Submit this script.

```
>cd $src/Runs
>chmod u+x run_name.sh
>queue=normal
>qsub -q $queue -o run_name.log run_name.sh
>qstat –u $USER
```

Debugging

The old debugging pages can be seen at: wiki:NAMEDebugging

Allinea DDT

We have not tried this yet. But VAR and the UM have found Allinea DDT useful. Paul Selwood has put together a user guide for the UM (Only available on Met Office Network).

Memory Profiling

Valgrind may be useful for memory profiling:

```
valgrind mpiexec -n 1 ./nameiiiDebug_64bit_par.exe
```

Windows

I have not tried this yet, but the following looks useful installing-mpi-for-windows

Things to be aware of when moving NAME versions

NAME v8.0

1. Met copy restore will only work with NAME using 1 rank. So you therefore need:

```
>mpiexec -n 1 ./nameiii_64bit_par.exe name_input_file.txt
```

2. Where is no benefit on running more than 1 rank per node (In fact the performance is degraded). Therefore if running NAME on a desktop or SPICE **use 1 rank**.
3. If using intel fortran you will need v17 or higher.
4. If running on the Met Office Cray "shared" queue you will need to:

```
>module load cray-snplauncher
```

Last modified on 05/06/19 09:53:13