

Hands-on: *INTI* Intel Haswell partition NPB-MZ-MPI / bt-mz_C.8

VI-HPS Team

Tutorial exercise objectives

- Familiarise with usage of VI-HPS tools
 - complementary tools' capabilities & interoperability
- Prepare to apply tools productively to *your* applications(s)
- Exercise is based on a small portable benchmark code
 - unlikely to have significant optimisation opportunities
- Optional (recommended) exercise extensions
 - analyse performance of alternative configurations
 - investigate effectiveness of system-specific compiler/MPI optimisations and/or placement/binding/affinity capabilities
 - investigate scalability and analyse scalability limiters
 - compare performance on different HPC platforms
 - ...

Compiler and MPI modules (INTI)

- Ensure that desired compiler and MPI modules (toolchain) are loaded first

```
% module list  
Currently Loaded Modulefiles:  
1) compil/x86_64 2) flavor/buildcompiler/intel/16 [...]   
16) mpi/openmpi/1.8.8
```

Alternatively switch compilers
(gcc) and/or MPI (intelmpi) ...

- Set-up and load the required VI-HPS tools modules (when needed)

```
% module load datadir/formation cube/4.3.5 scorep/3.1 (scalasca/2.3.1 vampir/9.3.0)  
load module datadir/formation (Data Directory)  
load module cube/4.3.5 (Cube)  
load module scorep/3.1 (Score-P)
```

... but then you **must** specify the
corresponding versions of tools!

- Copy tutorial sources to your \$SCRATCHDIR directory

```
% cd $SCRATCHDIR  
% tar zxvf $FORMATION_HOME/NPB3.3-MZ-MPI.tar.gz  
% cd NPB3.3-MZ-MPI
```

NPB-MZ-MPI Suite

- The NAS Parallel Benchmark suite (MPI+OpenMP version)

- Available from:

<http://www.nas.nasa.gov/Software/NPB>

- 3 benchmarks in Fortran77
- Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% ls
bin/      common/  jobscript/  Makefile  README.install  SP-MZ/
BT-MZ/   config/  LU-MZ/      README    README.tutorial  sys/
```

- Subdirectories contain source code for each benchmark
 - plus additional configuration and common code
- The provided distribution has already been configured for the tutorial, such that it is ready to “make” one or more of the benchmarks
 - but config/make.def may first need to be adjusted to specify appropriate compiler flags

NPB-MZ-MPI / BT: config/make.def

```
#           SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#-----
#-----
# Configured for generic MPI with compiler-specific OpenMP flags
#-----
#COMPFLAGS = -fopenmp           # GNU/GCC compiler
#COMPFLAGS = -openmp           # Intel14/15 compiler
COMPFLAGS = -qopenmp         # Intel compiler

...
#-----
# The Fortran compiler used for MPI programs
#-----
MPIF77 = mpif77

# Alternative variant to perform instrumentation
#MPIF77 = scorep --user mpif77

# Use PREP is a generic preposition for instrumentation preparation
#MPIF77 = $(PREP) mpif77
...
```

Uncomment flags specification
according to current compiler

Default (no instrumentation)

Hint: uncomment a compiler
wrapper to do instrumentation

Building an NPB-MZ-MPI Benchmark

```
% make
```

```
=====
=      NAS PARALLEL BENCHMARKS 3.3      =
=      MPI+OpenMP Multi-Zone Versions   =
=      F77                               =
=====
```

To make a NAS multi-zone benchmark type

```
make <benchmark-name> CLASS=<class> NPROCS=<nprocs>
```

```
where <benchmark-name> is "bt-mz", "lu-mz", or "sp-mz"
      <class>           is "S", "W", "A" through "F"
      <nprocs>         is number of processes
```

```
[...]
```

```
*****
* Custom build configuration is specified in config/make.def *
* Suggested tutorial exercise configuration for HPC systems: *
*      make bt-mz CLASS=C NPROCS=8                          *
*****
```

- Type "make" for instructions

Building an NPB-MZ-MPI Benchmark

```
% make bt-mz CLASS=C NPROCS=8
make[1]: Entering directory `BT-MZ'
make[2]: Entering directory `sys'
cc -o setparams setparams.c -lm
make[2]: Leaving directory `sys'
../sys/setparams bt-mz 8 C
make[2]: Entering directory `../BT-MZ'
mpif77 -c -O3 -qopenmp      bt.f
                               [...]
mpif77 -c -O3 -qopenmp      mpi_setup.f
cd ../common; mpif77 -c -O3 -qopenmp      print_results.f
cd ../common; mpif77 -c -O3 -qopenmp      timers.f
mpif77 -O3 -qopenmp -o ../bin/bt-mz_C.8 bt.o
  initialize.o exact_solution.o exact_rhs.o set_constants.o adi.o
  rhs.o zone_setup.o x_solve.o y_solve.o  exch_qbc.o solve_subs.o
  z_solve.o add.o error.o verify.o mpi_setup.o ../common/print_results.o
  ../common/timers.o
make[2]: Leaving directory `BT-MZ'
Built executable ../bin/bt-mz_C.8
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
 - benchmark name: **bt-mz**, lu-mz, sp-mz
 - the number of MPI processes: **NPROCS=8**
 - the benchmark class (S, W, A, B, C, D, E): **CLASS=C**

Shortcut: `% make suite`

NPB-MZ-MPI / BT (Block Tridiagonal Solver)

- What does it do?
 - Solves a discretized version of the unsteady, compressible Navier-Stokes equations in three spatial dimensions
 - Performs 200 time-steps on a regular 3-dimensional grid
- Implemented in 20 or so Fortran77 source modules

- Uses MPI & OpenMP in combination
 - 8 processes each with 8 threads should be reasonable for 2 compute nodes of INTI
 - bt-mz_B.8 should run in less than 4 seconds
 - bt-mz_C.8 should run in around 15 seconds

NPB-MZ-MPI / BT Reference Execution

```
% cd bin
% cp ../jobscript/inti/run.msub .
% less run.msub
% ccc_msub run.msub

% cat npb_btmz_<job_id>.oe
NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
Number of zones:      8 x      8
Iterations:  200      dt:  0.000300
Number of active processes:      8
Total number of threads:      48  (  6.0 threads/process)

Time step    1
Time step   20
  [...]
Time step  180
Time step  200
Verification Successful

BT-MZ Benchmark Completed.
Time in seconds = 18.78
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

Hint: save the benchmark output (or note the run time) to be able to refer to it later

Tutorial Exercise Steps

- Edit [config/make.def](#) to adjust build configuration
 - Modify specification of compiler/linker: [MPIF77](#)
- Make clean and then build new tool-specific executable

```
% make clean
% make bt-mz CLASS=C NPROCS=8
Built executable ../bin.scorep/bt-mz_C.8
```

- Change to the directory containing the new executable before running it with the desired tool configuration

```
% cd bin.scorep
% cp ../jobscript/inti/scorep.msub .
% ccc_msub scoprp.msub
```