



NPB3.3-MZ-MPI/BT tutorial example OpenMP+MPI application (IBM AIX/POE version)

Brian Wylie
Jülich Supercomputing Centre
b.wylie@fz-juelich.de
June 2012

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

- Connect using X11 forwarding

```
% ssh -X blizzard.dkrz.de
```

- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r /work/kg0166/tutorial/NPB3.3-MZ-MPI $WRKDIR
```

(Additional tutorial exercises can also be provided.)

- Load the latest Scalasca module

```
% module avail scalasca
scalasca/1.3.3-aixpoe-ibm          scalasca/1.4.2rc2-aixpoe-ibm
scalasca/1.4.1-aixpoe-ibm(default) scalasca/1.4.2rc3-aixpoe-ibm
% module load scalasca/1.4.2rc3-aixpoe-ibm
```

- Scalasca installation including interactive explorer GUI
 - configured for default MPI and IBM XL compilers
 - ▶ currently the only supported configuration for IBM POE systems
 - additional configurations would need to be installed for other MPI libraries and compiler suites
- Sample experiments available for examination

```
% ls /work/kg0166/tutorial/experiments/scalasca
README          epik_bt-mz_B_16x4_sum/  epik_bt-mz_B_16x4_trace/
                 epik_bt-mz_C_16x4_sum/  epik_bt-mz_C_16x4_trace/
```

- NAS Parallel Benchmark suite (sample MZ-MPI version)
 - Available from <http://www.nas.nasa.gov/Software/NPB>
 - 3 benchmarks (all in Fortran77, using OpenMP+MPI)
 - Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% cd NPB3.3-MZ-MPI; ls  
BT-MZ/  LU-MZ/  SP-MZ/  
bin/    common/ config/ jobscript/  Makefile  README  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's ready to “make” benchmarks and install them into a (tool-specific) “bin” subdirectory

- Type “make” for instructions

```
% make
```

```
=====
=      NAS Parallel Benchmarks 3.3      =
=      MPI+OpenMP Multi-Zone versions  =
=====
```

To make a NAS multi-zone benchmark type

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

To make a set of benchmarks, create the file config/suite.def according to the instructions in config/suite.def.template and type

```
make suite
```

```
*****
* Custom build configuration is specified in config/make.def *
* Suggested tutorial benchmark specification:                 *
* make bt-mz CLASS=B NPROCS=16                               *
*****
```

- Specify the benchmark configuration
 - benchmark name: **bt-mz**, lu-mz, sp-mz
 - the number of MPI processes: **NPROCS=16**
 - the benchmark class (S, W, A, B, C, D, E, F): **CLASS=B**

```
% make bt-mz CLASS=B NPROCS=16
cd BT-MZ; make CLASS=B NPROCS=16 VERSION=
gmake: Entering directory 'BT-MZ'
cd ../sys; cc -o setparams setparams.c -lm
../sys/setparams bt-mz 16 B
mpxlf_r -c -O -qsmp=omp bt.f
...
mpxlf_r -c -O -qsmp=omp setup_mpi.f
cd ../common; mpxlf_r -c -O -qsmp=omp print_results.f
cd ../common; mpxlf_r -c -O -qsmp=omp timers.f
mpxlf_r -O -qsmp=omp -o ../bin/bt-mz_B.16 \
    bt.o make_set.o initialize.o exact_solution.o exact_rhs.o \
    set_constants.o adi.o define.o copy_faces.o rhs.o solve_subs.o \
    x_solve.o y_solve.o z_solve.o add.o error.o verify.o setup_mpi.o \
    ../common/print_results.o ../common/timers.o
Built executable ../bin/bt-mz_B.16
gmake: Leaving directory 'BT-MZ'
```

- What does it do?
 - Solves a discretized version of unsteady, compressible Navier-Stokes equations in three spatial dimensions
 - Performs 200 time-steps on a regular 3-dimensional grid using ADI and verifies solution error within acceptable limit
 - Intra-zone computation with OpenMP, inter-zone with MPI
- Implemented in 20 or so Fortran77 source modules
- Runs with any number of MPI processes & OpenMP threads
 - bt-mz_B.16 x4 is reasonable for a POWER6 compute node
 - ▶ excess processes idle when run with more than compiled number
 - bt-mz_B.16 x4 should run in less than 10 seconds
 - ▶ typically runs more efficiently with more processes than threads
 - CLASS=C does four times the work and takes 4x longer

- Launch as an MPI application with OMP_NUM_THREADS set

```
% cd bin; OMP_NUM_THREADS=4 MP_PROCS=16 poe ./bt-mz_B.16
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:    16
```

```
Time step    1
Time step   20
Time step   40
Time step   60
Time step   80
Time step  100
Time step  120
Time step  140
Time step  160
Time step  180
Time step  200
Verification Successful
```

```
BT-MZ Benchmark Completed.
Time in seconds = 8.86
```

Hint: copy/edit example batch scripts from jobscript directory:
% llsubmit ../jobscript/run.ll

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

- The tutorial steps are similar and repeated for each tool
- Use the provided NPB3.3-MZ-MPI tutorial directory

```
% cd NPB3.3-MZ-MPI; ls
BT-MZ/  LU-MZ/  SP-MZ/
bin/    common/ config/ jobscript/  Makefile  README  sys/
```

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

```
% make clean
% make bt-mz CLASS=B NPROCS=16
Built executable ../bin.<<TOOL>>/bt-mz_B.16
```

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

```
% cd bin.<<TOOL>>
% llsubmit ../jobscript/run.ll
# OMP_NUM_THREADS=4 MP_PROCS=16 poe ./bt-mz_B.16
```

- config/make.def

```
#           SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS
#-----
# Items in this file may need to be changed for each platform.
...
OPENMP = -qsmp=omp # IBM XL
#-----
# The Fortran compiler used for hybrid MPI programs
#-----
MPIF77 = mpqlf_r
# Alternative variants to perform instrumentation
#MPIF77 = psc_instrument -t user,mpi mpqlf_r
#MPIF77 = scalasca -instrument mpqlf_r
#MPIF77 = tau_f90.sh
#MPIF77 = vtf77 -vt:hyb -vt:f77 mpqlf_r
# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpqlf_r
# This links MPI Fortran programs; usually the same as ${MPIF77}
FLINK   = $(MPIF77)
...
```

Adjust OMP flag if necessary

Default (no instrumentation)

Hint: uncomment one of these alternative compiler wrappers to perform instrumentation ...

... or this for generic variant

- Our tutorials are most often done with the Linux Live ISO so they use generic names
 - MPI compiler: `mpicc`, `mpicxx`, `mpif77`, `mpif90`
 - MPI launcher: `mpiexec -np ...`
 - OpenMP flag: `-fopenmp`
- If your system is different, then you need to adapt compiler/launch commands accordingly
- AIX POE/XL systems are certainly different
 - MPI compiler: `mpcc_r`, `mpCC_r`, `mpxlf_r`
 - MPI launcher: `poe ...`
 - OpenMP flag: `-qsmp=omp`