CP2K Best Practices for Intel® Cluster Ready

HPC ADVISORY COUNCIL
NETWORK OF EXPERTISE
BEST PRACTICES

1. Introduction:

The following best practices document is provided as courtesy of the HPC Advisory Council..

2. Application Description:

CP2K is a freely available (GPL) program, written in Fortran, to perform atomistic and molecular simulations of solid state, liquid, molecular and biological systems. It provides a general framework for different methods such as density functional theory (DFT) using a mixed Gaussian and plane waves approach (GPW), and classical pair and many-body potentials, solid state, liquid, molecular and biological systems. CP2K, including its sources and pointers to the authors can be found at:

http://cp2k.berlios.de/

3. Version Information:

Version of this date is used: CP2K version 2.2.188

4. Prerequisites:

4.1 Hardware:

The instructions from this best practice have been tested with the following configuration:

- Dell PowerEdge M610 38-node cluster
- Intel Xeon X5670 CPUs @ 2.93 MHz
- Memory: 24GB per node @ 1333MHz
- Mellanox ConnectX-2 QDR InfiniBand Adapters
- Mellanox QDR InfiniBand Switch

4.2 Software:

- Intel® Cluster Ready running RHEL 5.5
- Mellanox OFED 1.5.2 InfiniBand Software Stack
- Application: CP2K
- Compilers: Intel compilers
- MPI: Intel MPI 4, Open MPI 1.5.3, Platform MPI 8.0.1
- Benchmark workload: H2O-128.inp

5. Building CP2K

You need to define the Makefile that allows running CP2K in a parallel mode using MPI. The name of the file that you put is the name that you will use to compile the source code for your system architecture.

Changes in the source code

If you are running into an error below during compilation, you need to change the single quote to double quotes in the source file.

> <CP2K_ROOT>/makefiles/../src/fft_lib/fft_kinds.F:11: error: #include expects "FILENAME" or <FILENAME>

% vim <CP2K_ROOT> /cp2k/makefiles/../src/fft_lib/fft_kinds.F

Change this line:

> #include 'mkl_dfti.f90'

To this line:

> #include "mkl_dfti.f90"

Makefile for Intel Compilers with Intel MPIs

Here is an example to compile CP2K with Intel MPI with Intel MKL.:

<CP2K_ROOT>/arch/janus-ifort-impi.popt

> INTEL_MKL =/opt/intel/mkl
>
> INTEL_INC = $(INTEL_MKL)/include/fftw
>
> INTEL_LIB = $(INTEL_MKL)/lib/em64t
>
> CC = mpiicc
>
> FC = mpiifort
>
> LD = mpiifort
>
> AR = xiar –r
>
> DFLAGS = -D__INTEL -D__FFTSG -D__FFTW3 -D__FFTMKL -D__parallel -D__BLACS \
>
> -D__SCALAPACK
>
> CPPFLAGS = -C -traditional $(DFLAGS) -I$(INTEL_INC)
>
> FCFLAGS = $(DFLAGS) -I$(INTEL_INC) -O2 -xHost -heap-arrays 64 -fpp -free –unroll -pc 64\
>
> -convert big_endian
>
> LDFLAGS = $(FCFLAGS) -static-intel

MKL_LIB=/opt/intel/mkl/lib/em64t

LIBS = -L$(INTEL_LIB) -lfftw3xf_intel -lmkl_intel_lp64 -lmkl_sequential -lmkl_core\

-lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64

OBJECTS_ARCHITECTURE = machine_intel.o

Compiling BLACS for Platform MPI

Since Platform MPI does not have BLACS and ScaLaPACK libraries included in MKL, you will need to compile BLACS using the following instructions.

Download Scalapack installer 0.96, Scalapack 1.8.0 from this URL:

http://www.netlib.org/scalapack

Extract scalapack_installer_0.96 and run setup.py. The python script will download any necessary tgz packages automatically. For nodes that don't have internet connection, you will need to drop the tgz files to the "build" directory. Here is an example to compile using the scalapack installer:

% ./setup.py --f90 mpif90

At the end of the installation, you will get the static libraries compiled for BLAS, LAPACK, BLACS, and ScaLAPACK.

Makefile for Intel Compilers with Platform MPI

Here is an example to compile CP2K with Platform MPI.

<CP2K_ROOT>/arch/janus-ifort-pmpi.popt

INTEL_MKL =/opt/intel/mkl

INTEL_INC = $(INTEL_MKL)/include/fftw

INTEL_LIB = $(INTEL_MKL)/lib/em64t

BLACS_INC = /application/scalapack/scalapack_installer_0.96/include

BLACS_LIB = /application /scalapack/scalapack_installer_0.96/lib/libscalapack.a \

/application/scalapack/scalapack_installer_0.96/lib/blacsF77.a \

/application/scalapack/scalapack_installer_0.96/lib/blacs.a \

/application/scalapack/scalapack_installer_0.96/lib/blacsF77.a \

/application/scalapack/scalapack_installer_0.96/lib/libreflapack.a \

/application/scalapack/scalapack_installer_0.96/lib/librefblas.a

CC = mpicc

FC = mpif90

LD = mpif90

AR = ar –r

DFLAGS = -D__INTEL -D__FFTSG -D__FFTW3 -D__FFTMKL -D__parallel -D__BLACS -D__SCALAPACK

CPPFLAGS = -C -traditional $(DFLAGS) -I$(INTEL_INC) -I${BLACS_INC)

FCFLAGS = $(DFLAGS) -I$(INTEL_INC) -O2 -xHost -heap-arrays 64 -fpp -free -unroll -pc 64 -convert big_endian

LDFLAGS = $(FCFLAGS) -static-intel

MKL_LIB=/opt/intel/mkl/lib/em64t

LIBS = -I$(BLACS_INC) -L$(INTEL_LIB) -lfftw3xf_intel -lmkl_intel_lp64 -lmkl_sequential -lmkl_core $(BLACS_LIB)

OBJECTS_ARCHITECTURE = machine_intel.o

Makefile for Intel Compilers with Open MPI

Here is an example to compile CP2K with Open MPI and Intel MKL.

<CP2K_ROOT>/arch/janus-ifort-ompi.popt

INTEL_MKL =/opt/intel/mkl

INTEL_INC = $(INTEL_MKL)/include/fftw

INTEL_LIB = $(INTEL_MKL)/lib/em64t

CC = mpicc

FC = mpif90

LD = mpif90

AR = xiar –r

DFLAGS = -D__INTEL -D__FFTSG -D__FFTW3 -D__FFTMKL -D__parallel -D__BLACS -D__SCALAPACK

CPPFLAGS = -C -traditional $(DFLAGS) -I$(INTEL_INC)

FCFLAGS = $(DFLAGS) -I$(INTEL_INC) -O2 -xHost -heap-arrays 64 -fpp -free –unroll -pc 64 -convert big_endian

LDFLAGS = $(FCFLAGS) -static-intel

MKL_LIB=/opt/intel/mkl/lib/em64t

LIBS = -L$(INTEL_LIB) -lfftw3xf_intel -lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lmkl_scalapack_

lp64 -lmkl_blacs_openmpi_lp64

OBJECTS_ARCHITECTURE = machine_intel.o

Building CP2K

Using the name of the Makefile, you can then compile the source using the make command. For instance, use this command to compile using the arch/janus-ifort-ompi.opt Makefile file.

% cd <CP2K_ROOT>/makefiles

% make -j 4 ARCH=janus-ifort-ompi VERSION=popt

6. Running CP2K

The datasets for CP2K are located in the tests directory

Running with Intel MPI

% mpdboot --parallel-startup -r ssh -f < PATH_TO_HOSTFILE > -n 32

% mpiexec -np 456 -IB <CP2K_HOME>/exe/janus-ifort-ompi-mkl-blacs/cp2k.popt -i <CP2K_HOME>/tests/QS/benchmark/H2O-128.inp

%mpdallexit

Running with Platform MPI

% mpirun -np 384 -IBV -aff=automatic -prot -hostfile <PATH_TO_HOSTFILE> <CP2K_HOME>/exe/janus-ifort-ompi-mkl-blacs/cp2k.popt -i <CP2K_HOME>/tests/QS/benchmark/H2O-128.inp

Running with Open MPI

% mpirun -np 384 -mca btl self,sm,openib -hostfile < PATH_TO_HOSTFILE > -bind-to-core <CP2K_HOME>/exe/janus-ifort-ompi-mkl-blacs/cp2k.popt -i <CP2K_HOME>/tests/QS/benchmark/H2O-128.inp

HPC ADVISORY COUNCIL
NETWORK OF EXPERTISE

350 Oakmead Pkwy, Sunnyvale, CA 94085
Tel: 408-970-3400 • Fax: 408-970-3403
www.hpcadvisorycouncil.com