

<https://www.cp2k.org/>

6.1.0

- Intel Parallel Studio 2017 Update 8
- CUDA Toolkit 9.1.85 (only for GPU version)
- spglib/1.11.1

- cp2k-6.1.0.tar.gz
- plumed-2.4.3.tgz
- (以下のスクリプト内でもいくつかファイルを入手しています)
- install\_openmpi.patch

```

--- scripts/install_openmpi.sh.org 2018-11-20 10:08:10.000000000 +0900
+++ scripts/install_openmpi.sh 2018-11-20 10:07:47.000000000 +0900
@@ -56,13 +56,13 @@
    ;;
    __SYSTEM__)
        echo "===== Finding OpenMPI from system paths ====="
=====
-   check_command mpirun "openmpi"
-   check_command mpicc "openmpi"
-   check_command mpif90 "openmpi"
-   check_command mpic++ "openmpi"
-   check_lib -lmpi "openmpi"
-   add_include_from_paths OPENMPI_CFLAGS "mpi.h" $INCLUDE_PATHS
-   add_lib_from_paths OPENMPI_LDFLAGS "libmpi.*" $LIB_PATHS
+   #check_command mpirun "openmpi"
+   #check_command mpicc "openmpi"
+   #check_command mpif90 "openmpi"
+   #check_command mpic++ "openmpi"
+   #check_lib -lmpi "openmpi"
+   #add_include_from_paths OPENMPI_CFLAGS "mpi.h" $INCLUDE_PATHS
+   #add_lib_from_paths OPENMPI_LDFLAGS "libmpi.*" $LIB_PATHS
    ;;
    __DONTUSE__)
    ;;
@@ -90,24 +90,26 @@
    else
        mpi_bin=mpirun
    fi
-   # check openmpi version as reported by mpirun
-   raw_version=$(($mpi_bin --version 2>&1 | \
-       grep "(Open MPI)" | awk '{print $4}')
-   major_version=$(echo $raw_version | cut -d '.' -f 1)
-   minor_version=$(echo $raw_version | cut -d '.' -f 2)
-   # old versions required -lmpi_cxx to link cxx code, new version don't
-   if [ $major_version -gt 1 ]; then
+   ## check openmpi version as reported by mpirun
+   #raw_version=$(($mpi_bin --version 2>&1 | \
+   #       grep "(Open MPI)" | awk '{print $4}')
+   #major_version=$(echo $raw_version | cut -d '.' -f 1)
+   #minor_version=$(echo $raw_version | cut -d '.' -f 2)
+   ## old versions required -lmpi_cxx to link cxx code, new version don't
+   #if [ $major_version -gt 1 ]; then

```

```

OPENMPI_LIBS="-Impi"
- else
-   OPENMPI_LIBS="-Impi -Impi_cxx"
- fi
+ #else
+ # OPENMPI_LIBS="-Impi -Impi_cxx"
+ #fi
  # old versions didn't support MPI 3, so adjust __MPI_VERSION accordingly (needed e.g. for pexsi)
- if [ $major_version -lt 1 ] || \
-   [ $major_version -eq 1 -a $minor_version -lt 7 ] ; then
-   mpi2_dflags="-D__MPI_VERSION=2"
- else
+ #if [ $major_version -lt 1 ] || \
+ # [ $major_version -eq 1 -a $minor_version -lt 7 ] ; then
+ #   mpi2_dflags="-D__MPI_VERSION=2"
+ #else
  mpi2_dflags=""
- fi
+ #fi
+ OPENMPI_CFLAGS="-I${I_MPI_ROOT}/include64"
+ OPENMPI_LDFLAGS="-L${I_MPI_ROOT}/lib64 -Wl,-rpath=${I_MPI_ROOT}/lib64"
  cat <<EOF >> "${BUILDDIR}/setup_openmpi"
export OPENMPI_CFLAGS="${OPENMPI_CFLAGS}"
export OPENMPI_LDFLAGS="${OPENMPI_LDFLAGS}"

```

- install\_mkl\_intel\_intelmpi.patch

```

--- scripts/install_mkl.sh.org 2018-11-12 14:44:45.000000000 +0900
+++ scripts/install_mkl.sh 2018-11-12 14:50:40.000000000 +0900
@@ -66,7 +66,7 @@
  fi
  done
  # set the correct lib flags from MLK link adviser
- MKL_LIBS="-Wl,--start-group ${mkl_lib_dir}/libmkl_gf_lp64.a ${mkl_lib_dir}/
libmkl_core.a ${mkl_lib_dir}/libmkl_sequential.a"
+ MKL_LIBS="-lmkl_intel_lp64 -lmkl_sequential -lmkl_core"
  # check optional libraries
  if [ $MPI_MODE != no ] ; then
    enable_mkl_scalapack="__TRUE__"
@@ -74,11 +74,11 @@
  case $MPI_MODE in
    mpich)
      mkl_optional_libs="$mkl_optional_libs libmkl_blacs_lp64.a"
-      mkl_blacs_lib="libmkl_blacs_lp64.a"
+      mkl_blacs_lib="-lmkl_blacs_lp64"
      ;;
    openmpi)
      mkl_optional_libs="$mkl_optional_libs libmkl_blacs_openmpi_lp64
.a"
-      mkl_blacs_lib="libmkl_blacs_openmpi_lp64.a"
+      mkl_blacs_lib="-lmkl_blacs_intelmpi_lp64"
      ;;
  *)
    enable_mkl_scalapack="__FALSE__"
@@ -91,13 +91,13 @@
  done
  if [ $enable_mkl_scalapack = "__TRUE__" ] ; then
    echo "Using MKL provided ScaLAPACK and BLACS"
-    MKL_LIBS="${mkl_lib_dir}/libmkl_scalapack_lp64.a ${MKL_LIBS} ${mkl_lib_dir}/${mkl_blacs_lib}"
+    MKL_LIBS="-lmkl_scalapack_lp64 ${MKL_LIBS} ${mkl_blacs_lib}"
  fi
  else
    enable_mkl_scalapack="__FALSE__"
  fi
- MKL_LIBS="${MKL_LIBS} -Wl,--end-group -lpthread -lm -ldl"

```

```

- MKL_CFLAGS="$ {MKL_CFLAGS} -I$ {MKLROOT}/include"
+ MKL_LIBS="-L$ {mkl_lib_dir}/lib/intel64 $ {MKL_LIBS} -lpthread -lm"
+ MKL_CFLAGS="$ {MKL_CFLAGS} -I$ {MKLROOT}/include"

# write setup files
cat <<EOF > "$ {BUILDDIR}/setup_mkl"

```

- install\_elpa.patch

```

--- scripts/install_elpa.sh.org 2018-11-16 16:22:07.000000000 +0900
+++ scripts/install_elpa.sh 2018-11-16 16:29:20.000000000 +0900
@@ -92,6 +92,8 @@
    LDFLAGS="-Wl,--enable-new-dtags $ {MATH_LDFLAGS} $ {SCALAPACK_LDFLAGS} $ {cray_ldflags}" \
    LIBS="$ {SCALAPACK_LIBS} $ {MATH_LIBS}" \
    > configure.log 2>&1
+ sed -i -e "s/^wl=\\\"/wl=\\-Wl,\\\"/" \
+ -e "s/^pic_flag=\\\"/pic_flag=\\-fPIC\\\"/" libtool
make -j $NPROCS > make.log 2>&1
make install > install.log 2>&1
cd ..
@@ -114,6 +116,8 @@
    LDFLAGS="-Wl,--enable-new-dtags $ {MATH_LDFLAGS} $ {SCALAPACK_LDFLAGS} $ {cray_ldflags}" \
    LIBS="$ {SCALAPACK_LIBS} $ {MATH_LIBS}" \
    > configure.log 2>&1
+ sed -i -e "s/^wl=\\\"/wl=\\-Wl,\\\"/" \
+ -e "s/^pic_flag=\\\"/pic_flag=\\-fPIC\\\"/" libtool
make -j $NPROCS > make.log 2>&1
make install > install.log 2>&1
cd ..

```

- install\_quip.patch

```

--- scripts/install_quip.sh.org 2018-11-16 16:39:15.000000000 +0900
+++ scripts/install_quip.sh 2018-11-16 16:50:31.000000000 +0900
@@ -66,34 +66,24 @@
    -e "s|(cd build/.*)|1 >&- 2>&-|g" \
    bin/find_sizeof_fortran_t
fi
- sed -i \
- -e "s|(F77 *=\\).*|1 $ {FC}|g" \
- -e "s|(F90 *=\\).*|1 $ {FC}|g" \
- -e "s|(F95 *=\\).*|1 $ {FC}|g" \
- -e "s|(CC *=\\).*|1 $ {CC}|g" \
- -e "s|(CPLUSPLUS *=\\).*|1 $ {CXX}|g" \
- -e "s|(LINKER *=\\).*|1 $ {FC}|g" \
- -e "s|(FPP *=\\).*|1 $ {FC} -E -x f95-cpp-input|g" \
- -e "s|(QUIPPY_FCOMPILER *=\\).*|1 $ {FC}|g" \
- -e "s|(QUIPPY_CPP *=\\).*|1 $ {FC} -E -x f95-cpp-input|g" \
- arch/Makefile.linux_ $ {quip_arch}_gfortran
# enable debug symbols
- echo "F95FLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_gfo
rtran
- echo "F77FLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_gfo
rtran
- echo "CFLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_gfo
rtran
- echo "CPLUSPLUSFLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_gfortran
- export QUIP_ARCH=linux_ $ {quip_arch}_gfortran
+ echo "F95FLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_ifort_icc
+ echo "F77FLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_ifort_icc
+ echo "CFLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_ifort_icc
+ echo "CPLUSPLUSFLAGS += -g" >> arch/Makefile.linux_ $ {quip_arch}_ifort_icc
+ export QUIP_ARCH=linux_ $ {quip_arch}_ifort_icc
# hit enter a few times to accept defaults
- echo -e "$ {MATH_LDFLAGS} $ {MATH_LIBS} \\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n" | make config > configure.log
+ QUIP_MATH_LIBS="-lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lpthread -lm -ld"

```

```

+ echo -e "${MATH_LDFLAGS} ${QUIP_MATH_LIBS} \\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n\\n | make config > configure.log
# make -j does not work :(
make > make.log 2>&1
! [ -d "${pkg_install_dir}/include" ] && mkdir -p "${pkg_install_dir}/include"
! [ -d "${pkg_install_dir}/lib" ] && mkdir -p "${pkg_install_dir}/lib"
- cp build/linux_x86_64_gfortran/quip_unified_wrapper_module.mod \
+ cp build/linux_x86_64_ifort_icc/quip_unified_wrapper_module.mod \
"${pkg_install_dir}/include/"
- cp build/linux_x86_64_gfortran/*.a \
+ cp build/linux_x86_64_ifort_icc/*.a \
"${pkg_install_dir}/lib/"
- cp src/FoX-4.0.3/objs.linux_${quip_arch}_gfortran/lib/*.a \
+ cp src/FoX-4.0.3/objs.linux_${quip_arch}_ifort_icc/lib/*.a \
"${pkg_install_dir}/lib/"
cd ..
touch "${install_lock_file}"

```

## ビルド手順

```

#!/bin/sh

INSTDIR=/local/apl/lx/cp2k610
WORKDIR=/work/users/${USER}

VERSION=6.1.0

TARBALL=/home/users/${USER}/Software/CP2K/${VERSION}/cp2k-${VERSION}.tar.gz
#LIBGRID=/home/users/${USER}/Software/CP2K/${VERSION}/libgrid/cp2k-${VERSION}-i1708/tools/autotune_grid
PATCH_INST_OPENMPI=/home/users/${USER}/Software/CP2K/${VERSION}/install_openmpi.patch
PATCH_INST_MKL=/home/users/${USER}/Software/CP2K/${VERSION}/install_mkl_intel_intelmpi.patch
PATCH_INST_ELPA=/home/users/${USER}/Software/CP2K/${VERSION}/install_elpa.patch
PATCH_INST_QUIP=/home/users/${USER}/Software/CP2K/${VERSION}/install_quip.patch

PLUMED_VERSION=2.4.3
PLUMED_TARBALL=/home/users/${USER}/Software/PLUMED/${PLUMED_VERSION}/plumed-${PLUMED_VERSION}.tgz

PARALLEL=12

# NOTE: Some of header and library paths are defined in CPATH and
# LIBRARY_PATH environment variables from modulefiles.

#-----
umask 0022
export LANG=""
export LC_ALL=C

module purge
module load intel_parallelstudio/2017update8
module load spglib/1.11.1

cd $INSTDIR
if [ -d cp2k-${VERSION} ]; then
  mv cp2k-${VERSION} cp2k-erase
  rm -rf cp2k-erase &
fi

tar xzf ${TARBALL}
mv cp2k-${VERSION}/* .
rm -rf cp2k-${VERSION}/.{dockerignore,.gitignore}
rmdir cp2k-${VERSION}
#rm -rf tools/autotune_grid
#cp -rf ${LIBGRID} tools/autotune_grid
#cp ${LIBGRID}/libgrid.a tools/autotune_grid

# plumed

```

```

mkdir ${INSTDIR}/plumed
cd $WORKDIR
if [ -d plumed-${PLUMED_VERSION} ]; then
  mv plumed-${PLUMED_VERSION} plumed-erase
  rm -rf plumed-erase &
fi

tar xzf ${PLUMED_TARBALL}
cd plumed-${PLUMED_VERSION}
CC=mpiicc FC=mpiifort CXX=mpiicpc \
  ./configure --prefix=${INSTDIR}/plumed
make -j ${PARALLEL}
make check
make install

# cp2k toolchain
cd ${INSTDIR}/tools/toolchain
sed -e "s/blacs_openmpi/blacs_intelmpi/" scripts/install_mkl.sh > scripts/install_mkl.sh.impi
chmod 755 scripts/install_mkl.sh.impi

patch -p0 < $PATCH_INST_OPENMPI
patch -p0 < $PATCH_INST_MKL
patch -p0 < $PATCH_INST_ELPA
patch -p0 < $PATCH_INST_QUIP

CC=icc FC=ifort F77=ifort F90=ifort CXX=icpc \
MPICC=mpiicc MPICXX=mpiicpc MPIFC=mpiifort MPIF77=mpiifort MPIF90=mpiifort \
  ./install_cp2k_toolchain.sh --math-mode=mkl \
    --mpi-mode=openmpi \
    --with-elpa=install \
    --with-cmake=system \
    --with-mpich=no \
    --with-openmpi=system \
    --with-reflapack=no \
    -j ${PARALLEL}

# cheat the script
mv -f scripts/install_mkl.sh.impi scripts/install_mkl.sh

CC=icc FC=ifort F77=ifort F90=ifort CXX=icpc \
MPICC=mpiicc MPICXX=mpiicpc MPIFC=mpiifort MPIF77=mpiifort MPIF90=mpiifort \
  ./install_cp2k_toolchain.sh --math-mode=mkl \
    --mpi-mode=openmpi \
    --with-elpa=install \
    --with-ptscotch=install \
    --with-parmetis=install \
    --with-superlu=install \
    --with-pexsi=install \
    --with-quip=install \
    --with-cmake=system \
    --with-mpich=no \
    --with-openmpi=system \
    --with-reflapack=no \
    -j ${PARALLEL}

# LIBGRID_ESC=`echo ${INSTDIR}/tools/autotune_grid | sed -e 's/\//\\\\/g'`

# finish building toolchain, copy arch file to the proper place
## modify arch for cpu-only version
sed -e "/^CFLAGS    =/s/CFLAGS    =/CFLAGS    = -O2 /" \
  -e "/^FCFLAGS    =/s/FCFLAGS    =/FCFLAGS    = -O2 /" \
  -e "/^DFLAGS /s/\$/ -D__SPGLIB -D__PLUMED2/" \
  -e "/^LIBS /s/\$/ -lsympg -lz -lgsi -nofor_main/" \
  install/arch/local.psmpp > ${INSTDIR}/arch/rccs.psmpp
echo "include ${INSTDIR}/plumed/lib/plumed/src/lib/Plumed.inc" >> ${INSTDIR}/arch/rccs.psmpp

```

```

echo "EXTERNAL_OBJECTS=\$(PLUMED_STATIC_DEPENDENCIES)" >> ${INSTDIR}/arch/rccs.psmpp

## modify arch for cpu-gpu version
sed -e "/^CFLAGS    =/s/CFLAGS    =/CFLAGS    = -O2 /" \
    -e "/^FCFLAGS  =/s/FCFLAGS  =/FCFLAGS  = -O2 /" \
    -e "/^DFLAGS  /s/\$/ -D_SPGLIB -D_PLUMED2 -D_ACC -D_DBCSR_ACC -D_PW_CUDA/" \
    -e "/^LIBS  /s/\$/ -lsymspg -lz -lgsi -lcudart -lcufft -lcublas -nofor_main" \
    install/arch/local.psmpp > ${INSTDIR}/arch/rccs_cuda.psmpp
echo "include ${INSTDIR}/plumed/lib/plumed/src/lib/Plumed.inc" >> ${INSTDIR}/arch/rccs_cuda.psmpp
echo "EXTERNAL_OBJECTS=\$(PLUMED_STATIC_DEPENDENCIES)" >> ${INSTDIR}/arch/rccs_cuda.psmpp
echo "NVCC      = nvcc -D_GNUC__=5 -D_GNUC_MINOR__=3 -Xcompiler=--std=gnu++98" >> ${INSTDIR}/arch/rccs_cuda.psmpp
echo "NVFLAGS   = -gencode=arch=compute_60,code=sm_60 -gencode=arch=compute_70,code=sm_70 \$(DFLAGS)" >>
${INSTDIR}/arch/rccs_cuda.psmpp

#--

cd ${INSTDIR}/makefiles
make -j ${PARALLEL} ARCH=rccs VERSION=psmpp

module load cuda/9.1
make -j ${PARALLEL} ARCH=rccs_cuda VERSION=psmpp

```

## テスト

### plumed

インストールのログより。3つのエラーは全て軽微な数値エラー

```

+ check file ves/rt-td-vonmises/report.txt for more information
+ ERROR in test isdb/rt-emmi/
+ check file isdb/rt-emmi/report.txt for more information
+ ERROR in test isdb/rt-jcouplings-mi/
+ check file isdb/rt-jcouplings-mi/report.txt for more information
+ ERROR in test isdb/rt-jcouplings/
+ check file isdb/rt-jcouplings/report.txt for more information
+++++
+ Final report:
+ 248 tests performed, 161 tests not applicable
+ 3 errors found
+ Find the bug!
+ To replace references, go to the test directory and
+ type 'make reset'
+++++

```

### cp2k

テスト結果は /local/apl/lx/cp2k610/regtesting/rccs/psmpp もしくは /local/apl/lx/cp2k610/regtesting/rccs\_cuda/psmpp に残っています。

GPU版以外は ccfep 上でテストを実行しています。

```

#!/bin/sh

export LC_ALL=C
export LANG=""

module purge
module load intel_parallelstudio/2017update8
module load spglib/1.11.1

CP2K=/local/apl/lx/cp2k610/
CP2K_ARCH=rccs
CP2K_VER=psmpp
TIMEOUT=120
PARALLEL=16

ulimit -s unlimited

```

```
cd ${CP2K}/regtesting/${CP2K_ARCH}/${CP2K_VER}
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# serial test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 1 \
  -ompthreads 1 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
  -maxtasks ${PARALLEL} >& regtest_mpi1_omp1.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# omp test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 1 \
  -ompthreads 2 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
  -maxtasks ${PARALLEL} >& regtest_mpi1_omp2.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# mpi test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 2 \
  -ompthreads 1 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
  -maxtasks ${PARALLEL} >& regtest_mpi2_omp1.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# mpi/openmp test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 2 \
  -ompthreads 2 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
  -maxtasks ${PARALLEL} >& regtest_mpi2_omp2.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# yet another mpi test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 8 \
  -ompthreads 1 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
```

```
-maxtasks ${PARALLEL} >& regtest_mpi8_omp1.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}

# yet another mpi/openmp test
../../tools/regtesting/do_regtest \
  -nobuild \
  -nosvn \
  -arch ${CP2K_ARCH} \
  -version ${CP2K_VER} \
  -mpiranks 8 \
  -ompthreads 2 \
  -jobmaxtime ${TIMEOUT} \
  -cp2kdir ../../ \
  -maxtasks ${PARALLEL} >& regtest_mpi8_omp2.log
rm -rf LAST-${CP2K_ARCH}-${CP2K_VER}
```

#### ■ テスト結果: MPI1 - OMP1

```
----- Summary -----
Number of FAILED tests 1
Number of WRONG tests 0
Number of CORRECT tests 3008
Number of NEW tests 14
Total number of tests 3023
GREPME 1 0 3008 14 3023 X
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL

#### ■ テスト結果: MPI1 - OMP2

```
----- Summary -----
Number of FAILED tests 1
Number of WRONG tests 0
Number of CORRECT tests 3008
Number of NEW tests 14
Total number of tests 3023
GREPME 1 0 3008 14 3023 X
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL

#### ■ テスト結果: MPI2 - OMP1

```
----- Summary -----
Number of FAILED tests 1
Number of WRONG tests 0
Number of CORRECT tests 3053
Number of NEW tests 19
Total number of tests 3073
GREPME 1 0 3053 19 3073 X
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL

#### ■ テスト結果: MPI2 - OMP2

```
----- Summary -----
Number of FAILED tests 1
Number of WRONG tests 0
Number of CORRECT tests 3053
Number of NEW tests 19
Total number of tests 3073
GREPME 1 0 3053 19 3073 X
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL



## ■ テスト結果: MPI8 - OMP1

```
----- Summary -----  
Number of FAILED tests 8  
Number of WRONG tests 2  
Number of CORRECT tests 3012  
Number of NEW tests 16  
Total number of tests 3038  
GREPME 8 2 3012 16 3038 X  
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL
- QS/regtest-rma-3D/\* : RUNTIME FAIL

## ■ テスト結果: MPI8 - OMP2

```
----- Summary -----  
Number of FAILED tests 8  
Number of WRONG tests 2  
Number of CORRECT tests 3012  
Number of NEW tests 16  
Total number of tests 3038  
GREPME 8 2 3012 16 3038 X  
-----
```

- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL
- QS/regtest-rma-3D/\* : RUNTIME FAIL

## ■ テスト結果: GPU版 MPI2 - OMP2 (@ccca)

```
----- Summary -----  
Number of FAILED tests 3  
Number of WRONG tests 0  
Number of CORRECT tests 3052  
Number of NEW tests 19  
Total number of tests 3074  
GREPME 3 0 3052 19 3074 X  
-----
```

- QS/regtest-pao-2/H2O\_pao\_rotinv.inp : RUNTIME FAIL
- QS/regtest-rel/Hg\_rel.inp : RUNTIME FAIL
- QS/regtest-ri-mp2/opt\_basis\_O\_auto\_gen.inp : RUNTIME FAIL

## ベンチマーク

tests/QS/benchmark/H2O-64.inp を利用。(時間は grep "CP2K " \*.log で表示される値から)

jobtype	総コア数 (ノード数)	MPI	OMP	GPU	elapsed(sec)
core	18 (1)	18	1	-	77.148
small	40 (1)	40	1	-	45.537
small	80 (2)	80	1	-	36.421
small	160 (4)	160	1	-	27.490
small	160 (4)	80	2	-	33.158
small	160 (4)	32	5	-	27.090
gpu	12 (1)	12	1	1	130.869

## 雑多な情報

- 公式サイトのコパイラ対応情報 [https://www.cp2k.org/dev:compiler\\_support](https://www.cp2k.org/dev:compiler_support)
- libsmm は未検証(x86\_64 では xsmm で代用可能と判断)
- intel17-intelmpi とは異なり、intel17-openmpi3.1.0 では少しエラーが増える(mpi1\*omp1の条件でも)
  - 全体的に openmpi よりは intelmpi の方が正常に動作するように見える

- intel17-openmpi3.1.0-mkl2017.x の条件ではなぜか MPI 並列が異様に遅くなる
  - intel17-openmpi3.1.0-mkl2018.x ではなぜか MPI 並列の問題が解消する
- 今回のビルド条件では libgrid.a が OpenMP 並列時に正常に動作せず(ビルド時の -qopenmp の有無に関わらず)
  - intel17-openmpi3.1.0-mkl の場合ならば -qopenmp 付きでビルドした libgrid.a が正常に動作
  - libgrid.a は正常に動作する場合には 5% 程度の速度アップが望める
- intel18 ではエラーが少し増えるため今回は回避
- intel mpiの他バージョンについては未検証
- -O3 -xHost -ip の最適化条件で H2O-64 のベンチマークをおこなったところ、-O2 だけの方が高速だった
- GPU 版はかなり大きな系(>1000原子)でないと効率的ではないらしい