How Compiler GROMACS

Step #1:

Download GROMACS from GROMACS 5.0 from website:

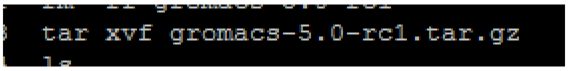
http://www.gromacs.org/Downloads

install reference guide : http://www.gromacs.org/Documentation/Installation_Instructions

Gromacs Userguide: http://www.gromacs.org/Documentation/Manual

Step #2:

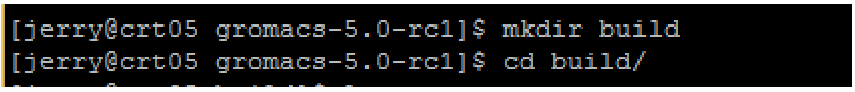
uncompess it by command " tar –xvf gromacs-<version>.tar.gz



```
tar xvf gromacs-5.0-rc1.tar.gz
```

Step #3:

create work directory by command: "mkdir build" and into the work directory – " cd build" as "cd <gromacs source code install directory>



```
[jerry@crt05 gromacs-5.0-rc1]$ mkdir build
[jerry@crt05 gromacs-5.0-rc1]$ cd build/
```

Step #3:

a. build cmake environment of gromacs to use Intel tools.

GROMACS 5.0.x need cmake 2.8+ (see **APPENDIX I**) and zlib (see **APPENDIX II**).

Set link library environment:

a) export ZLIB_DIR=<lib z install directory>

b) export MPICCDIR=<MPI install directory

b. Build cmake environment (for GROMACS compile'r parameter, please see **APPENDIX III**. And you can using a scripts to compiler gromacs – sample scripts see **APPENDIX IV**) of gromacs by command as followed:

For CPU version

cmake .. \

-DBUILD_SHARED_LIBS=OFF \

-DGMX_PREFER_STATIC_LIBS=ON \

-DGMX_BUILD_MDRUN_ONLY=ON \

-DGMX_FFT_LIBRARY=mkl \

-DCMAKE_INSTALL_PREFIX=<where the directory to install for gromacs> \

-DGMX_MPI=ON \

-DGMX_GPU=OFF \

-DGMX_XML=OFF \

-DGMX_SOFTWARE_INVSQRT=OFF \

-DGMX_SKIP_DEFAULT_CFLAGS=ON \

-DCMAKE_EXE_LINKER_FLAGS="-L${ZLIB_DIR}/lib64 -mkl=sequential" \

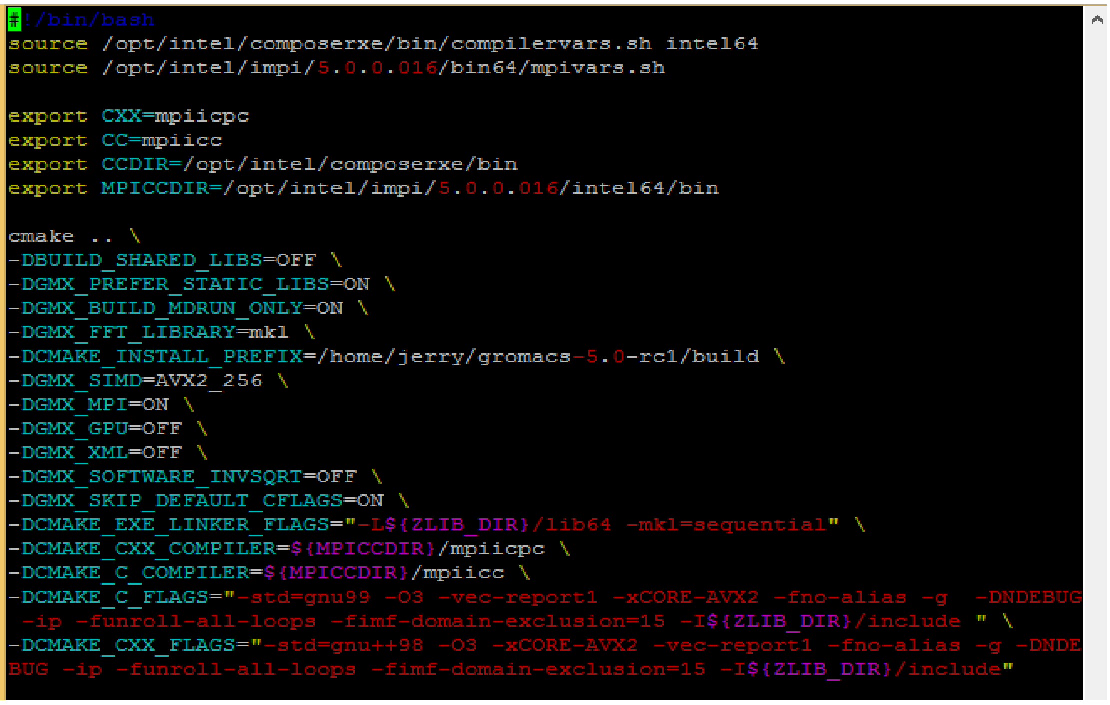
-DCMAKE_CXX_COMPILER=${MPICCDIR}/mpicxx \

-DCMAKE_C_COMPILER=${MPICCDIR}/mpicc \

-DCMAKE_C_FLAGS=" –cc=icc -std=gnu99 -O3 -vec-report1 -mmic -fno-alias -g -DNDEBUG -ip -funroll-all-loops -fimf-domain-exclusion=15 -I${ZLIB_DIR}/include " \

-DCMAKE_CXX_FLAGS="-cxx=icpc -std=gnu++98 -O3 -mmic -vec-report1 -fno-alias -g -DNDEBUG

-ip -funroll-all-loops -fimf-domain-exclusion=15 -I${ZLIB_DIR}/include"
"

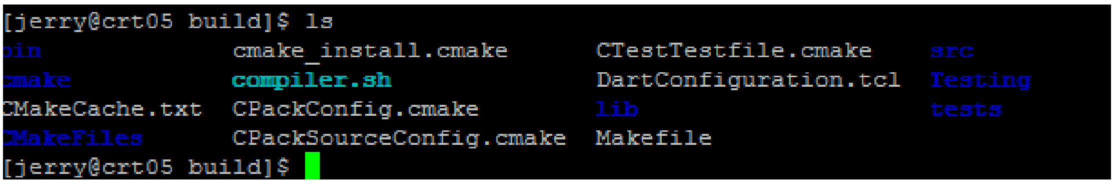


```
#!/bin/bash
source /opt/intel/composerxe/bin/compilervars.sh intel64
source /opt/intel/impi/5.0.0.016/bin64/mpivars.sh

export CXX=mpiicpc
export CC=mpiicc
export CCDIR=/opt/intel/composerxe/bin
export MPICCDIR=/opt/intel/impi/5.0.0.016/intel64/bin

cmake .. \
-DBUILD_SHARED_LIBS=OFF \
-DGMX_PREFER_STATIC_LIBS=ON \
-DGMX_BUILD_MDRUN_ONLY=ON \
-DGMX_FFT_LIBRARY=mkl \
-DCMAKE_INSTALL_PREFIX=/home/jerry/gromacs-5.0-rc1/build \
-DGMX_SIMD=AVX2_256 \
-DGMX_MPI=ON \
-DGMX_GPU=OFF \
-DGMX_XML=OFF \
-DGMX_SOFTWARE_INVSQRT=OFF \
-DGMX_SKIP_DEFAULT_CFLAGS=ON \
-DCMAKE_EXE_LINKER_FLAGS="-L${ZLIB_DIR}/lib64 -mkl=sequential" \
-DCMAKE_CXX_COMPILER=${MPICCDIR}/mpiicpc \
-DCMAKE_C_COMPILER=${MPICCDIR}/mpiicc \
-DCMAKE_C_FLAGS="-std=gnu99 -O3 -vec-report1 -xCORE-AVX2 -fno-alias -g  -DNDEBUG
 -ip -funroll-all-loops -fimf-domain-exclusion=15 -I${ZLIB_DIR}/include " \
-DCMAKE_CXX_FLAGS="-std=gnu++98 -O3 -xCORE-AVX2 -vec-report1 -fno-alias -g -DNDE
BUG -ip -funroll-all-loops -fimf-domain-exclusion=15 -I${ZLIB_DIR}/include"
```

Success to build gromacs compiler environment:



```
[jerry@crt05 build]$ ls
bin              cmake_install.cmake      CTestTestfile.cmake     src
cmake            compiler.sh              DartConfiguration.tcl   Testing
CMakeCache.txt   CPackConfig.cmake        lib                     tests
CMakeFiles       CPackSourceConfig.cmake  Makefile
[jerry@crt05 build]$
```

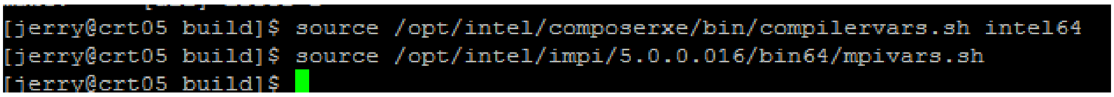
Step #4:

build Intel Tools environment before compiler.

BASH/KSH

source /opt/intel/composerxe/bin/compilervars.sh intel64

source /opt/intel/mkl/bin/mklvars.sh intel64

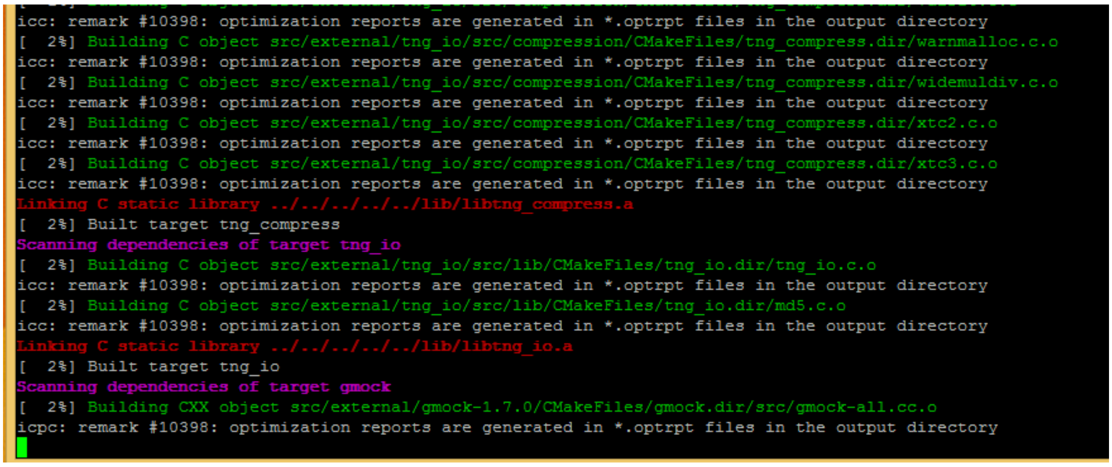


```
[jerry@crt05 build]$ source /opt/intel/composerxe/bin/compilervars.sh intel64
[jerry@crt05 build]$ source /opt/intel/impi/5.0.0.016/bin64/mpivars.sh
[jerry@crt05 build]$
```

Step #5:

Compiler gromacs by command “make –j 12”. Please see **APPENDIX V**, if got some error message.

Compilering:



```
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
[  2%] Building C object src/external/tng_io/src/compression/CMakeFiles/tng_compress.dir/warnmalloc.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
[  2%] Building C object src/external/tng_io/src/compression/CMakeFiles/tng_compress.dir/widemuldiv.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
[  2%] Building C object src/external/tng_io/src/compression/CMakeFiles/tng_compress.dir/xtc2.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
[  2%] Building C object src/external/tng_io/src/compression/CMakeFiles/tng_compress.dir/xtc3.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
Linking C static library ../../../../../lib/libtng_compress.a
[  2%] Built target tng_compress
Scanning dependencies of target tng_io
[  2%] Building C object src/external/tng_io/src/lib/CMakeFiles/tng_io.dir/tng_io.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
[  2%] Building C object src/external/tng_io/src/lib/CMakeFiles/tng_io.dir/md5.c.o
icc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
Linking C static library ../../../../../lib/libtng_io.a
[  2%] Built target tng_io
Scanning dependencies of target gmock
[  2%] Building CXX object src/external/gmock-1.7.0/CMakeFiles/gmock.dir/src/gmock-all.cc.o
icpc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
```

Step #6:

Install gromacs by command “make install”

How to run gromacs

Step #1:

if using Intel tools, build a running environment by command:

BASH/KSH

source /opt/intel/composerxe/bin/compilervars.sh intel64

source /opt/intel/mkl/bin/mklvars.sh intel64

if you native run gromacs on MIC, you can do on MIC as followed:”

source /opt/intel/mkl/bin/mklvars.sh mic

Step #2:

Create grocs input file *.tpr by command as followed:”

Grompp_mpi –f <mdp file> -n <ndx file> -p <top file> -c <data base. PDB file> -o <gromacs input file. *.tpr> [-maxwarn <number>]

“

Step #3:

run gromacs on cluster.

a. Run gromacs on Xeon by command:

mpirun –np <total mpi process number> -ppn <mpi process number per node> -machinefile <hostfiles> mdrun_mpi –v –deffnm <input files>

APPENDIX I: CMAKE INSTALL

Step #1:

Download cmake packets from website : http://www.cmake.org/

Reference guide form: http://www.cmake.org/cmake/help/documentation.html

Step #2:

Uncompress cmake packtes by command: tar –xvf cmake-<version>.tar.gz

Step #3:

Prepared to compiler cmake by command "./configure" using compiler which you choice. Set the install direction by arg: -prefix=<cmake install directory>.

a. If using GNU compiler, do nothing before "./configure"

b. If using Intel compiler, please set "CC, CXX, F77, F90 " to "icc, icpc. Ifort"

Step #4:

Compiler cmake.

a. If no previously version of cmake, please do command "./bootstrap ; make "

b. If already install cmake, please do command " make "

Step #4:

Install cmake on the system by command "make install", which will install cmake to "/usr/local/share/cmake-<version>". Replace old cmake command under /usr/local/bin/.

Step #5:

Relogin the server. Than you can using the new cmake.

APPENDIX II: Z LIRARTY INSTALL

Step #1:

Download zlib from website: http://www.zlib.net/

Step #2:

uncompress it by command:” tar xvf zlib-<version>.tgz”

Step #3:

configure it by command: “ ./configure –prefix=<zlib installed directory>” if you used Intel compiler, please do as followed:”

For Xeon:

Export CC=icc

Export CFLAGS=”-O3”

For Xeon Phi:

Export CC=icc

Export CFLAGS=”-O3 –mmic”

“

Step #4:

Compiler zlib by command “make all”.

Step #5:

Install Zlib by command: make install

APPENDIX III: GROMACS CMAKE PARAMETER

Detail see : **http://www.gromacs.org/Documentation/**

BUILD_SHARED_LIBS: enable | disable internal GROMACS library by ON | OFF.

GMX_PREFER_STATIC_LIBS: Enable | disable external (non- system) libraries by ON | OFF

GMX_BUILD_MDRUN_ONLY: Enable | disable if only build mdrun_mpi executable code by ON | OFF

GMX_FFT_LIBRARY: Choice which FFT library used between fftw,mkl,and fftpack. If used non mkl, please set **GMX_BUILD_OWN_FFTW**=on and download fftw packets. (fftpack already bundled with GROMACS source code).

CMAKE_INSTALL_PREFIX: where to install GROMACS

GMX_SIMD: support SIMD capability. Include

" SSE2,SSE4.1,AVX_128_FMA,AVX_256,AVX2_256,IBM_QPX,Sparc64_HPC_ACE, None". You can choice one for your HW architecture.

GMX_MPI: Enable | Disable MPI version of GROMACS by ON | OFF

GMX_GPU: Enable| Disable GPU Native model by ON | OFF.

GMX_SOFTWARE_INVSQRT: Enable | Disable to use SW sqrt by ON | OFF.

GMX_SKIP_DEFAULT_CFLAGS:

CMAKE_EXE_LINKER_FLAGS: Link flags which equal to the name of the C99.

CMAKE_CXX_COMPILER: CXX compiler which equal to the name of C++98.

CMAKE_C_COMPILER: C compiler

CMAKE_C_FLAGS: C compiler flags. Need support C99

CMAKE_CXX_FLAGS: CXX compiler flags. Need support C++98**.**

CMAKE_INSTALL_PREFIX: to install GROMACS to a non-standard location directory (default: /usr/local/gromacs)

GMX_DOUBLE: Enable | Disable running GROMACS with DP or SP by On | OFF

APPENDIX IV: COMPILER SAMPLING CRIPTS FILES

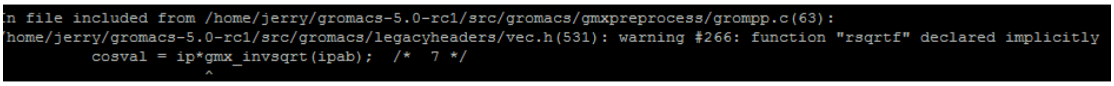
```
#!/bin/bash
source /opt/intel/composerxe/bin/compilervars.sh intel64
source /opt/intel/impi/5.0.0.016/bin64/mpivars.sh

export CXX=mpiicpc
export CC=mpiicc
export CCDIR=/opt/intel/composerxe/bin
export MPICCDIR=/opt/intel/impi/5.0.0.016/intel64/bin

cmake .. \
-DBUILD_SHARED_LIBS=OFF \
-DGMX_PREFER_STATIC_LIBS=ON \
-DGMX_BUILD_MDRUN_ONLY=ON \
-DGMX_FFT_LIBRARY=mkl \
-DCMAKE_INSTALL_PREFIX=/home/jerry/gromacs-5.0-rc1/build \
-DGMX_SIMD=AVX2_256 \
-DGMX_MPI=ON \
-DGMX_GPU=OFF \
-DGMX_XML=OFF \
-DGMX_SOFTWARE_INVSQRT=OFF \
-DGMX_SKIP_DEFAULT_CFLAGS=ON \
-DCMAKE_EXE_LINKER_FLAGS="-L${ZLIB_DIR}/lib64 -mkl=sequential" \
-DCMAKE_CXX_COMPILER=${MPICCDIR}/mpiicpc \
-DCMAKE_C_COMPILER=${MPICCDIR}/mpiicc \
-DCMAKE_C_FLAGS="-std=gnu99 -O3 -vec-report1 -xCORE-AVX2 -fno-alias -g   -DNDEBUG -ip -funroll-all-loops
-fimf-domain-exclusion=15 -I${ZLIB_DIR}/include " \
-DCMAKE_CXX_FLAGS="-std=gnu++98 -O3 -xCORE-AVX2 -vec-report1 -fno-alias -g -DNDEBUG -ip -funroll-all-loops
-fimf-domain-exclusion=15 -I${ZLIB_DIR}/include"
```

APPENDIX V: GROMACS COMPILER ERROR MESSAGE

a. vec.h(531): error: identifier "rsqrtf" is undefined cosval = ip*gmx_invsqrt(ipab); /* 7 */



```
n file included from /home/jerry/gromacs-5.0-rc1/src/gromacs/gmxpreprocess/grompp.c(63):
home/jerry/gromacs-5.0-rc1/src/gromacs/legacyheaders/vec.h(531): warning #266: function "rsqrtf" declared implicitly
        cosval = ip*gmx_invsqrt(ipab);  /*  7 */
                   ^
```

Solution A:

Modified file: build/src/config.h

Line#272:

#define HAVE_RSQRT ➔ /* #define HAVE_RSQRT */

Line#275:

#define HAVE_RSQRTF ➔ /* #define HAVE_RSQRTF */

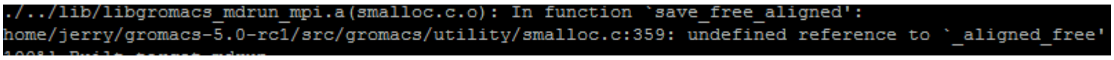
Solution B:

Before create gromacs makefile file, please do as followed:

Set BUILD_SHARED_LIBS=OFF to BUILD_SHARED_LIBS=ON

Set GMX_PREFER_STATIC_LIBS=ON to GMX_PREFER_STATIC_LIBS=OFF

b. smalloc.c:359: undefined reference to `_aligned_free'



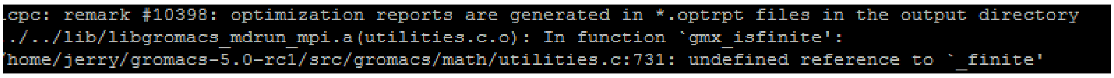
```
./../lib/libgromacs_mdrun_mpi.a(smalloc.c.o): In function `save_free_aligned':
home/jerry/gromacs-5.0-rc1/src/gromacs/utility/smalloc.c:359: undefined reference to `_aligned_free'
```

Solution:

Modified file build/src/config.h

Line#236: #define HAVE__ALIGNED_MALLOC ➔ /* #define HAVE__ALIGNED_MALLOC */

c. utilities.c:731: undefined reference to `_finite'



```
cpc: remark #10398: optimization reports are generated in *.optrpt files in the output directory
./../lib/libgromacs_mdrun_mpi.a(utilities.c.o): In function `gmx_isfinite':
home/jerry/gromacs-5.0-rc1/src/gromacs/math/utilities.c:731: undefined reference to `_finite'
```

Solution:

Modified file: <GROMACS Source Code directory>/src/gromacs/math/utilities.c

Line#731: returnval = _finite(x); ➔ /* returnval = _finite(x); */