CESM1.2.2 移植

——by 盖世女侠“边边”

系统

SUSE Linux Enterprise Server 11 (x86_64)

库的安装

除 ESMF 外，其他库超级推荐用 starman 安装，是**一键安装**哦！！

Starman 下载及安装参见 http://firststop-dongli.herokuapp.com/guides/1

例如，用 starman 安装 netcdf

进入 starman 目录下，

$starman shell　　##进入 starman 环境

$starman install netcdf -relax-env=LD_LIBRARY_PATH –verbose　##加上-verbose 可查看安装进程

##这样就一键安装好咯！！！ So Easy！！！

##netcdf3.6.2 及以后版本 netcdf 中 netcdf_c,netcdf_cxx 和 netcdf_fortran 是分开的哦，设置环境变量时（vi .bashrc）要注意， starman 安装的分别是 4.4.0、4.3.0 和 4.4.4 版本~

ESMF 库的安装

参考

http://www.earthsystemmodeling.org/esmf_releases/last_built/ESMF_usrdoc/node6.html

http://www.phy.pku.edu.cn/climate/wiki2/doku.php/misc/facility/climateserverinstall

在 program/packages 目录下，下载 esmf7_0_0

$wget

http://www.earthsystemmodeling.org/esmf_releases/non_public/ESMF_7_0_0/esmf_7_0_0_src.tar.gz

$gunzip esmf_7_0_0_src.tar.gz

$tar -xf esmf_7_0_0_src.tar

此时在 program/packages 目录下出现 esmf 目录~

然后设置环境变量~

$vi .bashrc

环境变量设置如下：

```
## NETCDF bianbian ##
export NETCDF=/home1/04310/bianqy/starset/software/netcdf/4.4.0/d9da11f9c8a28c4d37b834756233e684754eb87e
export PATH=$PATH:$NETCDF/bin
export MANPATH=$MANPATH:$NETCDF/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$NETCDF/lib
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$NETCDF/lib/pkgconfig

## LAPACK bianbian ##
export LAPACK=/home1/04310/bianqy/starset/software/lapack/3.6.1/0e3da623bb014bb78b9c3878a579c7cb228b0a31
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$LAPACK/lib64
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$LAPACK/lib64/pkgconfig

## HDF5 bianbian ##
export HDF5=/home1/04310/bianqy/starset/software/hdf5/1.8.16/d992a761327267855b825b15235ef99434ed9760
export PATH=$PATH:$HDF5/bin
export MANPATH=$MANPATH:$HDF5/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HDF5/lib

## CURL bianbian ##
export CURL=/home1/04310/bianqy/starset/software/curl/7.50.1/2b75c2844a82e4a8e92b409ec468205cf65223db
export PATH=$PATH:$CURL/bin
export MANPATH=$MANPATH:$CURL/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CURL/lib
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$CURL/lib/pkgconfig

## ESMF bianbian ##
export ESMF_OS=Linux
export ESMF_BOPT=g
export ESMF_PTHREADS=ON
export ESMF_OPENMP=ON
export ESMF_TESTMPMD=ON
export ESMF_DIR=/home1/04310/bianqy/program/packages/esmf
export ESMF_ABI=64
export ESMF_COMM=mpiuni
export ESMF_COMPILER=intel
export ESMF_TESTEXHAUSTIVE=ON
export ESMF_TESTHARNESS_ARRAY=RUN_ESMF_TestHarnessArrayUNI_2
export ESMF_TESTHARNESS_FIELD=RUN_ESMF_TestHarnessFieldUNI_1
export ESMF_NO_INTEGER_1_BYTE=FALSE
export ESMF_NO_INTEGER_2_BYTE=FALSE
export ESMF_CXXCOMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
export ESMF_CXXLINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
export ESMF_F90COMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
export ESMF_F90LINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
export ESMF_INSTALL_PREFIX=/home1/04310/bianqy/program/packages/esmf/../esmfinstall_dir
export ESMF_INSTALL_BINDIR=bin/bing/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_HEADERDIR=include
export ESMF_INSTALL_LIBDIR=lib/libg/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_MODDIR=mod/modg/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_DOCDIR=doc
```

备注：需要特别注意的是 ESMF_DIR 与 ESMF_INSTALL_PREFIX 的关系，安装路径 esmfinstall_dir 与 esmf 须同在 program/packages 目录下~~

设置好环境变量后，cd 到 esmf 目录下，

$make info ## 在安装前，使用该命令查看 ESMF 环境变量设置，可参考 http://www.earthsystemmodeling.org/esmf_releases/last_built/ESMF_usrdoc/node6.html

用户和最后的环境变量设置如下：

```
------------------------------------------------------------------
 * User set ESMF environment variables *
ESMF_OS=Linux
ESMF_OPENMP=ON
ESMF_TESTMPMD=ON
ESMF_TESTHARNESS_ARRAY=RUN_ESMF_TestHarnessArrayUNI_2
ESMF_PTHREADS=ON
ESMF_INSTALL_MODDIR=mod/modg/Linux.intel.64.mpiuni.default
ESMF_NO_INTEGER_2_BYTE=FALSE
ESMF_DIR=/home1/04310/bianqy/program/packages/esmf
ESMF_TESTHARNESS_FIELD=RUN_ESMF_TestHarnessFieldUNI_1
ESMF_NO_INTEGER_1_BYTE=FALSE
ESMF_INSTALL_BINDIR=bin/bing/Linux.intel.64.mpiuni.default
ESMF_INSTALL_LIBDIR=lib/libg/Linux.intel.64.mpiuni.default
ESMF_INSTALL_DOCDIR=doc
ESMF_COMM=mpiuni
ESMF_CXXCOMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
ESMF_INSTALL_PREFIX=/home1/04310/bianqy/program/packages/esmf/../esmfinstall_dir
ESMF_INSTALL_HEADERDIR=include
ESMF_CXXLINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
ESMF_F90COMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
ESMF_TESTEXHAUSTIVE=ON
ESMF_F90LINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
ESMF_BOPT=g
ESMF_ABI=64
ESMF_COMPILER=intel

------------------------------------------------------------------
 * ESMF environment variables *
ESMF_DIR: /home1/04310/bianqy/program/packages/esmf
ESMF_OS:                Linux
ESMF_MACHINE:           x86_64
ESMF_ABI:               64
ESMF_COMPILER:          intel
ESMF_BOPT:              g
ESMF_COMM:              mpiuni
ESMF_SITE:              default
ESMF_PTHREADS:          ON
ESMF_OPENMP:            ON
ESMF_OPENACC:           OFF
ESMF_ARRAY_LITE:        FALSE
ESMF_NO_INTEGER_1_BYTE: FALSE
ESMF_NO_INTEGER_2_BYTE: FALSE
ESMF_FORTRANSYMBOLS:    default
ESMF_DEFER_LIB_BUILD:   ON
ESMF_SHARED_LIB_BUILD:  ON
ESMF_TESTEXHAUSTIVE:    ON
ESMF_TESTWITHTHREADS:   OFF
ESMF_TESTMPMD:          ON
ESMF_TESTSHAREDOBJ:     OFF
ESMF_TESTFORCEOPENMP:   OFF
ESMF_TESTFORCEOPENACC:  OFF
ESMF_TESTHARNESS_ARRAY: RUN_ESMF_TestHarnessArrayUNI_2
ESMF_TESTHARNESS_FIELD: RUN_ESMF_TestHarnessFieldUNI_1
ESMF_MPIRUN:            /home1/04310/bianqy/program/packages/esmf/src/Infrastructure/stubs/mpiuni/mpirun
ESMF_MPIMPMDRUN:
```

上述网址中环境变量设置如下：

```
------------------------------------------------------------
 * User set ESMF environment variables *
ESMF_OS=Linux
ESMF_TESTMPMD=ON
ESMF_TESTHARNESS_ARRAY=RUN_ESMF_TestHarnessArrayUNI_2
ESMF_DIR=/nobackupp10/scvasque/daily_builds/intel/esmf
ESMF_TESTHARNESS_FIELD=RUN_ESMF_TestHarnessFieldUNI_1
ESMF_TESTWITHTHREADS=OFF
ESMF_COMM=mpiuni
ESMF_INSTALL_PREFIX= /nobackupp10/scvasque/daily_builds/intel/esmf/.. \
        /install_dir
ESMF_TESTEXHAUSTIVE=ON
ESMF_BOPT=g
ESMF_SITE=default
ESMF_ABI=64
ESMF_COMPILER=intel

------------------------------------------------------------
 * ESMF environment variables *
ESMF_DIR: /nobackupp10/scvasque/daily_builds/intel/esmf
ESMF_OS:                Linux
ESMF_MACHINE:           x86_64
ESMF_ABI:               64
ESMF_COMPILER:          intel
ESMF_BOPT:              g
ESMF_COMM:              mpiuni
ESMF_SITE:              default
ESMF_PTHREADS:          ON
ESMF_OPENMP:            ON
ESMF_ARRAY_LITE:        FALSE
ESMF_NO_INTEGER_1_BYTE: FALSE
ESMF_NO_INTEGER_2_BYTE: FALSE
ESMF_FORTRANSYMBOLS:    default
ESMF_DEFER_LIB_BUILD:   ON
ESMF_TESTEXHAUSTIVE:    ON
ESMF_TESTWITHTHREADS:   OFF
ESMF_TESTMPMD:          ON
ESMF_TESTSHAREDOBJ:     OFF
ESMF_TESTFORCEOPENMP:   OFF
ESMF_TESTHARNESS_ARRAY: RUN_ESMF_TestHarnessArrayUNI_2
ESMF_TESTHARNESS_FIELD: RUN_ESMF_TestHarnessFieldUNI_1
ESMF_MPIRUN:            /nobackupp10/scvasque/daily_builds/intel/esmf/src/ \
                         Infrastructure/stubs/mpiuni/mpirun

------------------------------------------------------------
 * ESMF environment variables pointing to 3rd party software *

------------------------------------------------------------
 * ESMF environment variables for final installation *
ESMF_INSTALL_PREFIX:    /nobackupp10/scvasque/daily_builds/intel/esmf/../ \
        install_dir
ESMF_INSTALL_HEADERDIR: include
ESMF_INSTALL_MODDIR:    mod/modg/Linux.intel.64.mpiuni.default
ESMF_INSTALL_LIBDIR:    lib/libg/Linux.intel.64.mpiuni.default
ESMF_INSTALL_BINDIR:    bin/bing/Linux.intel.64.mpiuni.default
ESMF_INSTALL_DOCDIR:    doc
```

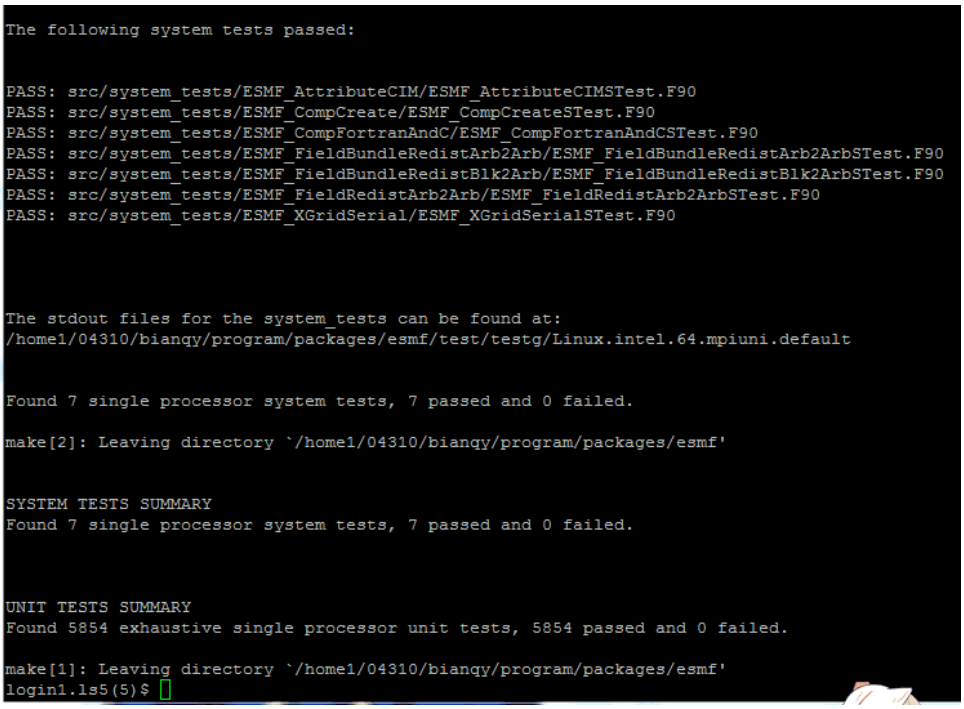
然后

$make

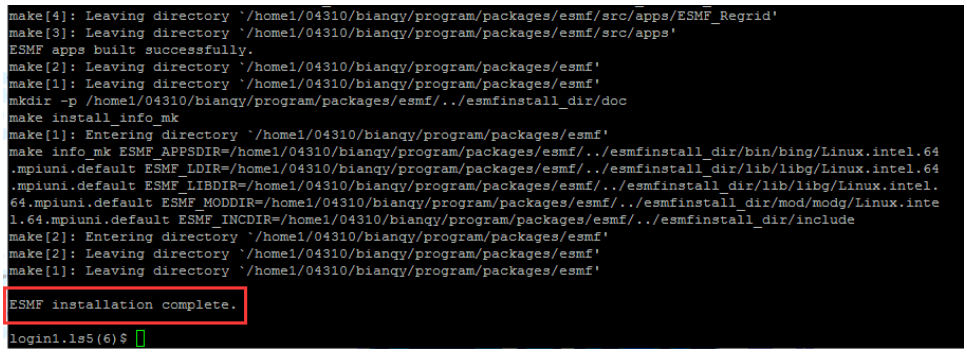
$make check

$make install

在 make check 之后，check 的结果会告诉你是否通过了各种各样的检查，若出现下图所示，则编译成功~~



make install 安装好/bin /include /lib /mod 和/doc 后，安装成功，如下图所示：



CESM1_2_2 下载

参考

http://www.cesm.ucar.edu/models/cesm1.2/tags/index.html

下载方法一：（该方法 svn 下载部分文件出错，其中有 pio，因为 googlecode 链接失效，参见 https://bb.cgd.ucar.edu/pio-error-cesm122 和 http://bbs.06climate.com/forum.php?mod=viewthread&tid=47465 ）

在 cesm 目录下，

$svn co --username guestuser --password [password]

https://svn-ccsm-models.cgd.ucar.edu/cesm1/release_tags/cesm1_2_2 cesm1_2_2

然后提示需要输入密码，例如 friendly，然后提示输入用户名，用户名是默认的，只能输入 guestuser

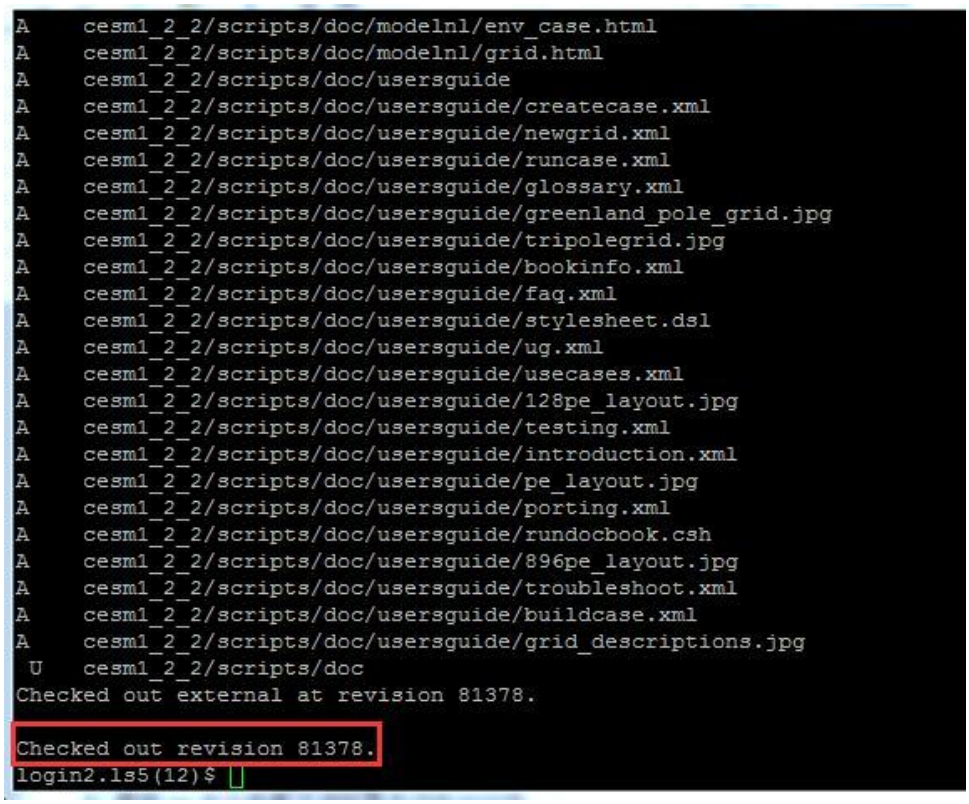
系统提示：store password unencrypted (yes/no)? 输入 yes

备注：密码不宜太复杂密码不宜太复杂密码不宜太复杂~~~~~~重要的事说三遍~~~~~~~~~~

下载成功的话，最后一行会有提示

Checked out revision xxxxx

如图：



下载方法二：

QQ 群

气象家园 NCAR/CESM&CLM&CAM （245364236）

群文件 CESM1_2_2 完整安装包~

来来来运行个测试 MPI 的小程序先

Userguide Chapter 5 Porting Overview “Hello World”

$cd mpi_test （在这个文件夹下测试，这个是自己定的）

$vi fhello_world_mpi.F90 （创建 fhello_world_mpi.F90）

然后按照 userguide 把那些命令敲进去就好啦~~（注意：程序第一行把“.F90”去掉，主要是把“.”去掉，不然编译出错~~）

最后如图：

```
program fhello_world_mpi
  use mpi
  implicit none
  integer ( kind = 4 ) error
  integer ( kind = 4 ) id
  integer p
  character(len=MPI_MAX_PROCESSOR_NAME) :: name
  integer clen
  integer, allocatable :: mype(:)
  real ( kind = 8 ) wtime

  call MPI_Init ( error )
  call MPI_Comm_size ( MPI_COMM_WORLD, p, error )
  call MPI_Comm_rank ( MPI_COMM_WORLD, id, error )
  if ( id == 0 ) then

    wtime = MPI_Wtime ( )

    write ( *, '(a)' ) ' '
    write ( *, '(a)' ) 'HELLO_MPI - Master process:'
    write ( *, '(a)' ) '  FORTRAN90/MPI version'
    write ( *, '(a)' ) ' '
    write ( *, '(a)' ) '  An MPI test program.'
    write ( *, '(a)' ) ' '
    write ( *, '(a,i8)' ) '  The number of processes is ', p
    write ( *, '(a)' ) ' '
  end if

  call MPI_GET_PROCESSOR_NAME(NAME, CLEN, ERROR)

  write ( *, '(a)' ) ' '
  write ( *, '(a,i8,a,a)' ) '  Process ', id, ' says "Hello, world!" ',name(1:cle
n)

  call MPI_Finalize ( error )

end program
```

$mpif90 fhello_world_mpi.F90 ##编译

编译后生成可执行文件 a.out

$mpirun ./a.out （我的是$ibrun ./a.out，因为没有 mpi~）

如图：

```
nid00009(1)$ ls
a.out  fhello_world_mpi.F90
nid00009(2)$ ibrun ./a.out
TACC: Starting up job 489863
TACC: Starting parallel tasks...

HELLO_MPI - Master process:
  FORTRAN90/MPI version

  An MPI test program.

  The number of processes is       24


  Process        0 says "Hello, world!" nid00009

  Process        1 says "Hello, world!" nid00009

  Process        5 says "Hello, world!" nid00009

  Process        9 says "Hello, world!" nid00009

  Process       10 says "Hello, world!" nid00009

  Process       13 says "Hello, world!" nid00009

  Process       16 says "Hello, world!" nid00009

  Process       19 says "Hello, world!" nid00009

  Process       20 says "Hello, world!" nid00009

  Process       21 says "Hello, world!" nid00009
```

运行成功，就可以移植啦~~

CESM1_2_2 移植

参考

http://blog.csdn.net/a1333888/article/details/51346876

移植 CESM121 到南信大的大型机.docx

一、环境变量设置

①vi .bashrc （这是我的环境变量设置，主要是 netcdf）

```
# NETCDF bianbian ##
export NETCDF=/home1/04310/bianqy/starset/software/netcdf/4.4.0/d9da11f9c8a28c4d37b834756233e684754eb87e
export PATH=$PATH:$NETCDF/bin
export MANPATH=$MANPATH:$NETCDF/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$NETCDF/lib
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$NETCDF/lib/pkgconfig

## LAPACK bianbian ##
export LAPACK=/home1/04310/bianqy/starset/software/lapack/3.6.1/0e3da623bb014bb78b9c3878a579c7cb228b0a31
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$LAPACK/lib64
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$LAPACK/lib64/pkgconfig

## HDF5 bianbian ##
export HDF5=/home1/04310/bianqy/starset/software/hdf5/1.8.16/d992a761327267855b825b15235ef99434ed9760
export PATH=$PATH:$HDF5/bin
export MANPATH=$MANPATH:$HDF5/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HDF5/lib

## CURL bianbian ##
export CURL=/home1/04310/bianqy/starset/software/curl/7.50.1/2b75c2844a82e4a8e92b409ec468205cf65223db
export PATH=$PATH:$CURL/bin
export MANPATH=$MANPATH:$CURL/share/man
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CURL/lib
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$CURL/lib/pkgconfig

## ESMF bianbian ##
export ESMF_OS=Linux
export ESMF_BOPT=g
export ESMF_PTHREADS=ON
export ESMF_OPENMP=ON
export ESMF_TESTMPMD=ON
export ESMF_DIR=/home1/04310/bianqy/program/packages/esmf
export ESMF_ABI=64
export ESMF_COMM=mpiuni
export ESMF_COMPILER=intel
export ESMF_TESTEXHAUSTIVE=ON
export ESMF_TESTHARNESS_ARRAY=RUN_ESMF_TestHarnessArrayUNI_2
export ESMF_TESTHARNESS_FIELD=RUN_ESMF_TestHarnessFieldUNI_1
export ESMF_NO_INTEGER_1_BYTE=FALSE
export ESMF_NO_INTEGER_2_BYTE=FALSE
export ESMF_CXXCOMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
export ESMF_CXXLINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/icpc
export ESMF_F90COMPILER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
export ESMF_F90LINKER=/opt/apps/intel/16.0.1.150/compilers_and_libraries_2016.1.150/linux/bin/intel64/ifort
export ESMF_INSTALL_PREFIX=/home1/04310/bianqy/program/packages/esmf/../esmfinstall_dir
export ESMF_INSTALL_BINDIR=bin/bing/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_HEADERDIR=include
export ESMF_INSTALL_LIBDIR=lib/libg/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_MODDIR=mod/modg/Linux.intel.64.mpiuni.default
export ESMF_INSTALL_DOCDIR=doc
```

②module load 软件名/版本号（也可以在.bashrc 里设置，这儿采用的是直接在命令提示界面分别运行 module load ***，最后 module save，这样下次登录时所有 module 已经自动 load 了，不需要额外更改.bashrc，要查看已经 load 的 module，运行 module list 即可~~用 module spider 可查看有哪些可以 load 的 module~~【跟赵龙师兄学的，哈哈~】）

$module load ncl_ncarg/6.3.0

$module load nco/4.5.4

$module load matlab/2015b

$module load parallel-netcdf/4.3.3.1

$module load cray_mpich/7.3.0

$module load intel/16.0.1

$module cmake/3.4.1

$module save

```
login1.ls5(7)$ module list

Currently Loaded Modules:
  1) TACC/1.0       3) intel/16.0.1       5) ncl_ncarg/6.3.0   7) parallel-netcdf/4.3.3.1
  2) matlab/2015b   4) cray_mpich/7.3.0   6) nco/4.5.4         8) cmake/3.4.1

login1.ls5(8)$
```

二、配置

需要配置/cesm1_2_2/scripts/ccsm_utils/Machines 目录下四个文件，

①vi config_machines.xml（根据自己的情况酌情修改，其中//machine name 自己取的，saber；目录是自己定的，注意要提前创建出来；DIN_LOC_ROOT 和 DIN_LOC_ROOT_CLMFORC 路径里不能有变量名，否则会出错；）

```
<machine MACH="saber">
        <DESC>generic linux (intel), batch system is SGE</DESC>                    <!-- can be a
nything -->
        <OS>LINUX</OS>                          <!-- LINUX,Darwin,CNL,AIX,BGL,BGP -->
        <COMPILERS>intel</COMPILERS>     <!-- intel,ibm,pgi,pathscale,gnu,cray,lahey -->
        <MPILIBS>mpich</MPILIBS>              <!-- openmpi, mpich, ibm, mpi-serial -->
        <CESMSCRATCHROOT>/scratch/04310/bianqy</CESMSCRATCHROOT>                  <!-- complete path to th
e 'scratch' directory -->
        <RUNDIR>/scratch/04310/bianqy/CESM1_2_2/case/$CASE/run</RUNDIR>                  <!-- complete p
ath to the run directory -->
        <EXEROOT>/scratch/04310/bianqy/CESM1_2_2/case/$CASE/bld</EXEROOT>                <!-- complete p
ath to the build directory -->
        <DIN_LOC_ROOT>/scratch/04310/bianqy/inputdata/CESM1_2_2/inputdata</DIN_LOC_ROOT>          <!-- compl
ete path to the inputdata directory -->
        <DIN_LOC_ROOT_CLMFORC>/scratch/04310/bianqy/inputdata/CESM1_2_2/inputdata/lmwg</DIN_LOC_ROOT_CLMFORC>
 <!-- path to the optional forcing data for CLM (for CRUNCEP forcing) -->
        <DOUT_S>TRUE</DOUT_S>                                        <!-- logical for short term archivin
g -->
        <DOUT_S_ROOT>/work/04310/bianqy/lonestar/CESM1_2_2/archive/$CASE</DOUT_S_ROOT>              <!-- com
plete path to a short term archiving directory -->
        <DOUT_L_MSROOT>USERDEFINED_optional_run</DOUT_L_MSROOT>          <!-- complete path to a long term a
rchiving directory -->
        <CCSM_BASELINE>/scratch/04310/bianqy/inputdata/CESM1_2_2/ccsm_baseline</CCSM_BASELINE>          <!--
 where the cesm testing scripts write and read baseline results -->
        <CCSM_CPRNC>/home1/04310/bianqy/cesm1_2_2/cesm1_2_2/tools/cprnc</CCSM_CPRNC>               <!-- path
 to the cprnc tool used to compare netcdf history files in testing -->
        <BATCHQUERY>USERDEFINED_optional_run</BATCHQUERY>
        <BATCHSUBMIT>USERDEFINED_optional_run</BATCHSUBMIT>
        <SUPPORTED_BY>USERDEFINED_optional</SUPPORTED_BY>
        <GMAKE_J>1</GMAKE_J>
        <MAX_TASKS_PER_NODE>12</MAX_TASKS_PER_NODE>
</machine>
```

② vi config_compiler.xml，进行修改（把 mach name 改成自己取的机器名；如果环境变量里有 MPI 的话，此处要添加 MPI_PATH 路径； PNETCDF_PATH 参考赵龙师兄的设置，因为是 module load 的~ 所以说如果有在同一台服务器上安装 CESM 的师兄师姐的话，可以参考他们这四个文件的设置，会省时省力很多 O(∩_∩)O~~）

```
<compiler MACH="saber">
  <NETCDF_PATH>/home1/04310/bianqy/starset/software/netcdf/4.4.0/d9da11f9c8a28c4d37b834756233e684754eb87
e</NETCDF_PATH>
  <PNETCDF_PATH>$(TACC_PNETCDF_DIR)</PNETCDF_PATH>
  <ADD_SLIBS>-L/home1/04310/bianqy/starset/software/netcdf/4.4.0/d9da11f9c8a28c4d37b834756233e684754eb87
e/lib -lnetcdf -lnetcdff</ADD_SLIBS>
  <ADD_CPPDEFS></ADD_CPPDEFS>
  <CONFIG_ARGS></CONFIG_ARGS>
  <ESMF_LIBDIR></ESMF_LIBDIR>
  <MPI_LIB_NAME>mpich</MPI_LIB_NAME>
  <MPI_PATH></MPI_PATH>
</compiler>
```

③执行 cp env_mach_specific.userdefined env_mach_specific.saber （就是把文件名后缀 userdefined 改成自己的机器名）
然后 vi env_mach_specific.saber，按照如下添加（同样，如果环境变量里有 MPI 的话，此处要添加 MPI_PATH 路径等，路径同 config_compiler.xml 的设置）

```
#source /opt/modules/default/init/csh
#if ( $COMPILER == "pgi" ) then
#  module load pgi
#endif
#module load netcdf

#limit coredumpsize unlimited
#set MPI_PATH=/opt/apps/intel16/cray_mpich/7.3.0
set NETCDF_PATH=/home1/04310/bianqy/starset/software/netcdf/4.4.0/d9da11f9c8a28c4d37b834756233e684754eb8
7e
#setenv INC_MPI ${MPI_PATH}/include
#setenv LIB_MPI ${MPI_PATH}/lib
setenv INC_NETCDF ${NETCDF_PATH}/include
setenv LIB_NETCDF ${NETCDF_PATH}/lib
```

④执行 cp mkbatch.userdefined mkbatch.saber（同样是改机器名字），
然后 vi mkbatch.saber 按照自己机器的情况修改，主要改动有两处，就是绿框中的内容~
第一，由于我的服务器提交作业方式既不是#PBS 也不是#BSUB，而是#SBATCH，因此 自己添加的#SBATCH，且添加的内容必须紧跟#!/bin/csh –f 之下（不知道这是不是#SBATCH 的特殊要求~~）
第二，mpiexec 和 mpirun 根据自己的服务器选一个，去掉前面的“#”即可，但由于我使用的服务器既没有 mpiexec，也没有 mpirun，是 ibrun，因此用 ibrun 替换了 mpirun（在此记录里特地感谢下 NCAR CESM Software Engineer, Jim Edwards，发邮件请教了他这个问题，秒回~~）
最后的设置如下：

```
#!/bin/csh -f
#SBATCH -J ${jobname}        # job name
#SBATCH -o $HOME/JOB_output/${jobname}.o%j  # output and error file name (%j expands t
o jobID)
#SBATCH -n ${ntasks}         # total number of mpi tasks requested
#SBATCH -N ${nodes}          # number of nodes requested
#SBATCH -p development       # queue (partition) -- normal, development, etc.
#SBATCH -t 01:30:00          # run time (hh:mm:ss) - 1.5 hours

#limit coredumpsize 1000000
#limit stacksize unlimited


EOF1

#-------------------------------------------------------------------------------
else if ($PHASE == set_exe) then
#-------------------------------------------------------------------------------

set maxthrds = `${CASEROOT}/Tools/taskmaker.pl -maxthrds`
set maxtasks = `${CASEROOT}/Tools/taskmaker.pl -sumtasks`

cat >> ${CASEROOT}/${CASE}.run << EOF1
sleep 25
cd \$RUNDIR
echo "\`date\` -- CSM EXECUTION BEGINS HERE"

setenv OMP_NUM_THREADS ${maxthrds}

#=============================================================================
# USERDEFINED
# edit job launching
#=============================================================================

#mpiexec -n ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID
ibrun -np ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID

wait
echo "\`date\` -- CSM EXECUTION HAS FINISHED"

EOF1
```

运行测试

进入 cesm1_2_2/scrips 文件夹，

$ cd ccsm4/scripts

$ create_newcase -case test \

-res f19_g16 \

-compset X \

-mach saber

(test 是自己定的 case 的名字，mach 后是自己定义的机器名，成功后会生成一个名为 test 的文件夹)

$cd test

$./cesm_setup

$./test.build

$sbatch test.run （成功后在<RUNDIR>路径下会有.nc 文件输出，可以通过查看 test 文件夹下边一个叫 CaseStatus 的文件看看是不是运行成功，或 test/run 下面的 cesm.log….或 $HOME/JOB_output，这个是自己在 mkbatch 文件中设置的~~）

模式验证（Port Validation）

./create_test -testname ERS.f19_g16.X -mach saber -compiler intel -testid t01

cd ERS.f19_g16.X.saber_intel.t01/

./ERS.f19_g16.X.saber_intel.t01.build

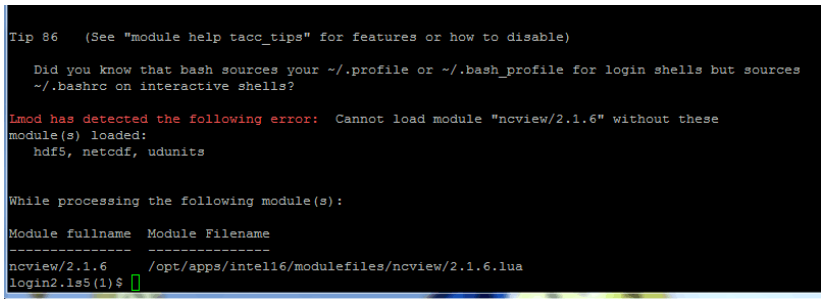
sbatch ERS.f19_g16.X.saber_intel.t01.test

vi TestStatus

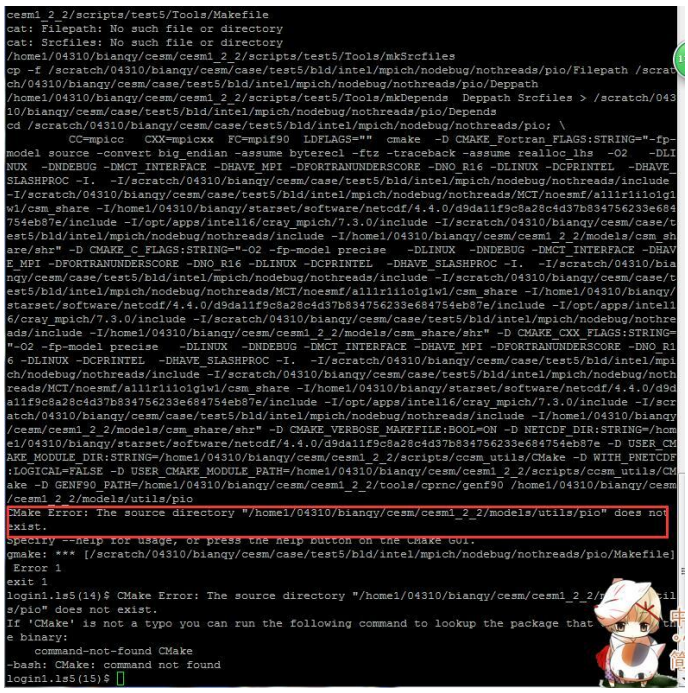
vi TestStatus.out

移植期间遇到的问题

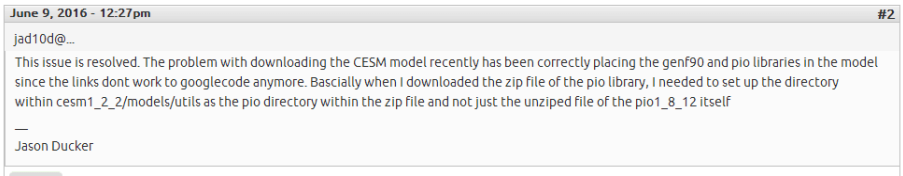
Q1：



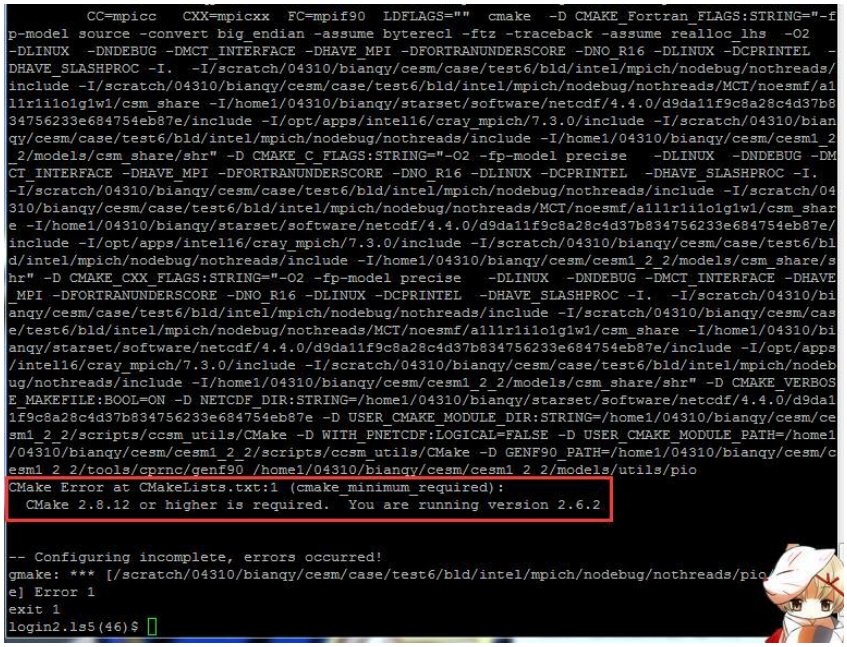
Q2：



A2：这个是官网下载的 CESM1_2_2 有问题导致的~~

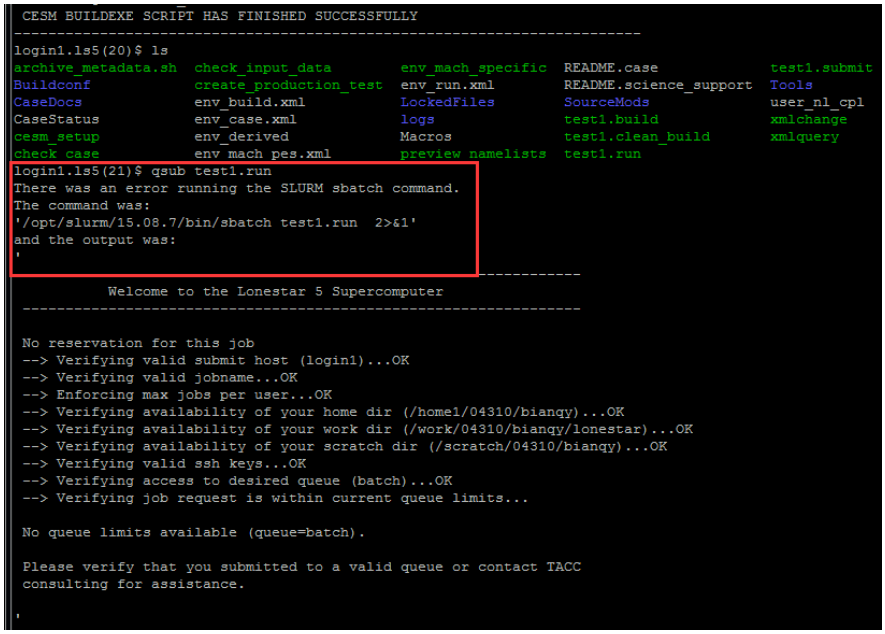


Q3：



A3：遇到这个问题时是自己在 github 上下的 ParallelIO-master，然后放在 models/utils 文件夹下，换了 CESM1_2_2 的安装包后，没遇到过这个问题~~再者，这个 ParallelIO-master 文件本身与 CESM1_2_2 也不兼容~~

Q4：



A4：出错时我的 mkbatch.saber 设置是这样的，

```
env_mach_specific.edison      mkbatch.brutus                mkbatch.yellowstone
env_mach_specific.eos         mkbatch.eastwind              mkDepends
env_mach_specific.erebus      mkbatch.edison                mkSrcfiles
env_mach_specific.evergreen   mkbatch.eos                   README
env_mach_specific.gaea        mkbatch.erebus                taskmaker.pl
login1.ls5(29)$ vi mkbatch.saber
# USERDEFINED
# This is where the batch submission is set.  The above code computes
# the total number of tasks, nodes, and other things that can be useful
# here.  Use PBS, BSUB, or whatever the local environment supports.
#===========================================================================

#PBS -N ${jobname}
#PBS -q ${qname}
#PBS -l nodes=${nodes}:ppn=${taskpernode}
#PBS -l walltime=${tlimit}
#PBS -r n
#PBS -j oe
#PBS -S /bin/csh -V

##BSUB -l nodes=${nodes}:ppn=${taskpernode}:walltime=${tlimit}
##BSUB -q ${qname}
###BSUB -k eo
###BSUB -J $CASE
###BSUB -W ${tlimit}

#limit coredumpsize 1000000
#limit stacksize unlimited


EOF1

#############################################################################
else if ($PHASE == set_exe) then
#############################################################################

set maxthrds = `${CASEROOT}/Tools/taskmaker.pl -maxthrds`
set maxtasks = `${CASEROOT}/Tools/taskmaker.pl -sumtasks`

cat >> ${CASEROOT}/${CASE}.run << EOF1
sleep 25
cd \$RUNDIR
echo "\`date\` -- CSM EXECUTION BEGINS HERE"

setenv OMP_NUM_THREADS ${maxthrds}

#===========================================================================
# USERDEFINED
# edit job launching
#===========================================================================

#mpiexec -n ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID
mpirun -np ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID

wait
echo "\`date\` -- CSM EXECUTION HAS FINISHED"

EOF1
```

同样的服务器，赵龙师兄的设置是这样的，而且没有 mpirun 或 mpiexec，

```
#!/bin/csh -f
#SBATCH -J ${jobname}      # job name
#SBATCH -o $HOME/JOB_output/${jobname}.o%j  # output and error file name (%j expands to jobID)
#SBATCH -n ${ntasks}       # total number of mpi tasks requested
#SBATCH -N ${nodes}        # number of nodes requested
#SBATCH -p development     # queue (partition) -- normal, development, etc.
#SBATCH -t 01:30:00        # run time (hh:mm:ss) - 1.5 hours
#SBATCH --mail-user=username@tacc.utexas.edu
#SBATCH --mail-type=begin  # email me when the job starts
#SBATCH --mail-type=end    # email me when the job finishes

#limit coredumpsize 1000000
#limit stacksize unlimited
```

最后拷贝赵龙师兄的设置添加到我的 mkbatch.saber 文件中，如下

```
#--- Job name is first fifteen characters of case name ---
set jobname = `echo ${CASE} | cut -c1-15`

if ($?TESTMODE) then
 set file = $CASEROOT/${CASE}.test
else
 set file = $CASEROOT/${CASE}.run
endif

cat >! $file << EOF1
#!/bin/csh -f
#=============================================================================
# USERDEFINED
# This is where the batch submission is set.  The above code computes
# the total number of tasks, nodes, and other things that can be useful
# here.  Use PBS, BSUB, or whatever the local environment supports.
#=============================================================================

##PBS -N ${jobname}
##PBS -q ${qname}
##PBS -l nodes=${nodes}:ppn=${taskpernode}
##PBS -l walltime=${tlimit}
##PBS -r n
##PBS -j oe
##PBS -S /bin/csh -V

##BSUB -l nodes=${nodes}:ppn=${taskpernode}:walltime=${tlimit}
##BSUB -q ${qname}
###BSUB -k eo
###BSUB -J $CASE
###BSUB -W ${tlimit}

#SBATCH -J ${jobname}      # job name
#SBATCH -o $HOME/JOB_output/${jobname}.o%j  # output and error file name (%j expands to jobID)
#SBATCH -n ${ntasks}       # total number of mpi tasks requested
#SBATCH -N ${nodes}        # number of nodes requested
#SBATCH -p development     # queue (partition) -- normal, development, etc.
#SBATCH -t 01:30:00        # run time (hh:mm:ss) - 1.5 hours

#limit coredumpsize 1000000
#limit stacksize unlimited


EOF1
```

又出错，这是因为#SBATCH 等设置必须紧邻#!/bin/csh –f，

Submit a batch job with sbatch

Use Slurm's sbatch command to submit a job. Specify the resources needed for your job (e.g., number of nodes/tasks needed, job run time) in a Slurm job script. See "/share/doc/slurm" for example Slurm job submission scripts.

```
login1$ sbatch myjobscript
```

where "myjobscript" is the name of a UNIX format text file containing job script commands. This file can contain both shell commands and special statements that include #SBATCH options and resource specifications; shell commands other than the initial parser line (e.g. #!/bin/bash) must follow all #SBATCH Slurm directives. Some of the most common options are described in Table 4 below and in the example job scripts. Details are available online in man pages (e.g., execute "man sbatch" on Lonestar 5).

最后的设置如下：

```
#!/bin/csh -f
#SBATCH -J ${jobname}        # job name
#SBATCH -o $HOME/JOB_output/${jobname}.o%j  # output and error file name (%j expands t
o jobID)
#SBATCH -n ${ntasks}         # total number of mpi tasks requested
#SBATCH -N ${nodes}          # number of nodes requested
#SBATCH -p development       # queue (partition) -- normal, development, etc.
#SBATCH -t 01:30:00          # run time (hh:mm:ss) - 1.5 hours

#limit coredumpsize 1000000
#limit stacksize unlimited


EOF1

#-------------------------------------------------------------------------
else if ($PHASE == set_exe) then
#-------------------------------------------------------------------------

set maxthrds = `${CASEROOT}/Tools/taskmaker.pl -maxthrds`
set maxtasks = `${CASEROOT}/Tools/taskmaker.pl -sumtasks`

cat >> ${CASEROOT}/${CASE}.run << EOF1
sleep 25
cd \$RUNDIR
echo "\`date\` -- CSM EXECUTION BEGINS HERE"

setenv OMP_NUM_THREADS ${maxthrds}

#----------------------------------------------------------------------------
# USERDEFINED
# edit job launching
#----------------------------------------------------------------------------

#mpiexec -n ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID
ibrun -np ${maxtasks} \$EXEROOT/cesm.exe >&! cesm.log.\$LID

wait
echo "\`date\` -- CSM EXECUTION HAS FINISHED"

EOF1
```

Q5：

```
ls5.tacc.utexas.edu - PuTTY
================================================================================
[ ^[[1;31mERROR^[[0m ] You have invoked an unsupported MPI job launch command:
[ ^[[1;31mERROR^[[0m ]   /opt/apps/tacc/bin/mpirun
[ ^[[1;31mERROR^[[0m ] Lonestar5 uses the ibrun MPI job launcher.
[ ^[[1;31mERROR^[[0m ] For more information on appropriate ibrun command options,
[ ^[[1;31mERROR^[[0m ] please visit our user guide here:
[ ^[[1;31mERROR^[[0m ] ^[[1;32mhttps://portal.tacc.utexas.edu/user-guides/lonestar5^[[0m
================================================================================
```

A5：这个是因为我的服务器处理 MPI job（不会翻译的说。。并行计算任务？？）是 ibrun，而我在 mkbatch 文件中选的是 mpirun，因此出错，最后将 mpirun 直接替换为了 ibrun。

Q6：

```
ls5.tacc.utexas.edu - PuTTY
--------------------------------------------------------------------------------
 CESM BUILDNML SCRIPT STARTING
 - To prestage restarts, untar a restart.tar file into /scratch/04310/bianqy/CESM1_2_2/case/test4/
run
 infile is /home1/04310/bianqy/cesm1_2_2/cesm1_2_2/scripts/test4/Buildconf/cplconf/cesm_namelist
 CESM BUILDNML SCRIPT HAS FINISHED SUCCESSFULLY
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
 CESM PRESTAGE SCRIPT STARTING
 - Case input data directory, DIN_LOC_ROOT, is /scratch/04310/bianqy/inputdata/CESM1_2_2/inputdata
 - Checking the existence of input datasets in DIN_LOC_ROOT
 CESM PRESTAGE SCRIPT HAS FINISHED SUCCESSFULLY
--------------------------------------------------------------------------------
Wed Nov  2 22:42:40 CDT 2016 -- CSM EXECUTION BEGINS HERE
Wed Nov  2 22:42:40 CDT 2016 -- CSM EXECUTION HAS FINISHED
ls: No match.
Model did not complete - no cesm.log file present - exiting
```

A5：这个问题出现的原因就是遇到 Q5 后，我直接把 mkbatch 中 mpirun 和 mpiexec 前都加上了“#”，也就是保持他们最初的样子，就遇到这个问题了，所以说 mpirun 或者 mpiexec

这个是必须设置滴~~~~~~~~