# CESM Research Tools: CLM4 in CESM1.1.1 User's Guide Documentation

**Erik Kluzek**
**NCAR**
**Jun-26-2013**
**$URL: https://svn-ccsm-models.cgd.ucar.edu/clm2/branches/cesm1_1_0_rel/models/lnd/clm/doc/UsersGuide/clm_ug.xr**
**$**

**CESM Research Tools: CLM4 in CESM1.1.1 User's Guide Documentation**

by Erik Kluzek

The user's guide to CLM4 in CESM1.1.1 which is the active land surface model component of CESM1.1.1. The purpose of this guide is to instruct both the novice and experienced user, as well as CLM developers in the use of CLM4 for land-surface climate modeling.

# Dedication

Dedicated to the Land Model Working Group, winners of the 2008 CCSM Distinguished Achievement Award. May you continue to collaborate together well, and continue to drive the science of land surface modeling forward with your diligent and persistent efforts.

# Table of Contents


**Acknowledgments** ..... **viii**

**Introduction** ..... **ix**

**Introduction to the CLM4 User's Guide** ..... **x**

**Important Notes and Best Practices for Usage of CLM4** ..... **xii**

**How to Use This Document** ..... **xiii**

**What is new with CLM4 in CESM1.1.1 since previous public releases?** ..... **xiv**

**Quickstart to using CLM4** ..... **xv**

**What is scientifically validated and functional in CLM4 in CESM1.1.1?** ..... **xvii**

Standard Configuration and Namelist Options that are Validated ..... xvii

Configure Modes NOT scientifically validated, documented, supported or, in some cases, even advised to be used: ..... xvii

Namelist options that should NOT be exercised: ..... xvii

Build-Namelist options that should NOT be exercised: ..... xviii

Namelist items that should NOT be exercised: ..... xviii

**What are the UNIX utilities required to use CLM?** ..... **xix**

**Other resources to get help from** ..... **xx**

The CESM User's-Guide ..... xx

The CESM Bulletin Board ..... xx

The CLM web pages ..... xx

Reporting bugs in CLM4 ..... xxi

Some Acronymn's We'll be Using ..... xxi

**1. How to customize the configuration for a case with CLM** ..... **1**

Choosing a compset using CLM ..... 1

Compsets coupled to data atmosphere and stub ocean/sea-ice ("I" compsets) ..... 1

Compsets coupled to active atmosphere with data ocean ..... 3

Fully coupled compsets with fully active ocean, sea-ice, and atmosphere ..... 3

Conclusion to choosing a compset ..... 3

Customizing the CLM setup ..... 3

CLM Script configuration items ..... 4

User Namelist ..... 10

Precedence of Options ..... 11

Setting Your Initial Conditions File ..... 12

Doing a hybrid simulation to provide initial conditions ..... 12

Doing a branch simulation to provide initial conditions ..... 13

Providing a finidat file in your `user_nl_clm` file ..... 13

Adding a finidat file to the XML database ..... 13

Other noteworthy configuration items ..... 13

Downloading DATM Forcing Data ..... 15

Customizing via the build script files ..... 16

More information on the CLM **configure** script ..... 16

Help on CLM **configure** ..... 16

Customizing the CLM namelist ..... 17

Definition of Namelist items and their default values ..... 17

Definition of CLM history variables ..... 18
Examples of using different namelist features ..... 64
The default namelist ..... 64
Adding/removing fields on your primary history file ..... 65
Adding auxiliary history files and changing output frequency ..... 65
Removing all history fields ..... 66
Various ways to change history output averaging flags ..... 66
Outputting history files as a vector in order to analyze the plant function types within gridcells ..... 67
Conclusion to namelist examples ..... 68
Customizing the DATM Namelist and Streams files ..... 69
`CLM_QIAN` mode and it's DATM settings ..... 71
`CLM1PT` mode and it's DATM settings ..... 71
`CPLHIST3HrWx` mode and it's DATM settings ..... 71
Conclusion to customizing chapter ..... 72
**2. Using the CLM tools to create your own input datasets ..... 73**
Common environment variables and options used in building the FORTRAN tools ..... 73
General information on running the FORTRAN tools ..... 77
Running FORTRAN tools with namelists ..... 77
Running FORTRAN tools with command line options ..... 78
Running FORTRAN tools built with SMP=TRUE ..... 78
Using NCL scripts ..... 78
The File Creation Process ..... 79
Using the **cprnc** tool to compare two history files ..... 88
Using **interpinic** to interpolate initial conditions to different resolutions ..... 89
Creating an output SCRIP grid file at a resolution to run the model on ..... 90
Using **mknocnmap.pl** to create grid and maps for single-point regional grids ..... 90
Creating mapping files that **mksurfdata_map** will use ..... 91
Creating a domain file for CLM and DATM ..... 92
Creating a set of regional datasets from existing global datasets ..... 96
Using mksurfdata_map to create surface datasets from grid datasets ..... 96
Running **mksurfdata.pl** ..... 96
Running **mksurfdata_map** by Hand ..... 98
Preparing your **mksurfdata_map** namelist ..... 98
Single Point options to **mksurfdata_map** ..... 103
Standard Practices when using **mksurfdata_map** ..... 103
Converting unstructured grid output to gridded datasets for post processing ..... 104
How to Customize Datasets for particular Observational Sites ..... 105
Conclusion of tools description ..... 105
**3. Adding New Resolutions or New Files to the build-namelist Database ..... 106**
Managing Your Own Data-files ..... 106
Adding Resolution Names ..... 108
Adding or Changing Default Filenames ..... 108
What are the required files? ..... 109

**4. How to run some special cases ..... 112**
Running with the prognostic crop model on ..... 112
Running with the irrigation model on ..... 113
Spinning up the Satellite Phenology Model (CLMSP spinup) ..... 113
Spinning up the biogeochemistry Carbon-Nitrogen Model (CN spinup) ..... 114
Spinning up the Carbon-Nitrogen Dynamic Global Vegetation Model (CNDV spinup) ..... 115
Running with MOAR data as atmospheric forcing to spinup the model ..... 116
Running with your own previous simulation as atmospheric forcing to spinup the model ..... 117
Running stand-alone CLM with transient historical $CO_2$ concentration ..... 118
**5. How to run Single-Point/Regional cases ..... 122**
Which Single Point Option Should I choose? ..... 122
Running PTS_MODE configurations ..... 123
Warning about Running with a Single-Processor on a Batch Machine ..... 124
Running Supported Single-point/Regional Datasets ..... 124
Running Supported Single-point Datasets that have their own Atmospheric Forcing ..... 125
Creating your own single-point/regional surface datasets ..... 127
Using **getregional_datasets.pl** to get a complete suite of single-point/regional surface datasets from global ones ..... 129
Running with your own atmosphere forcing ..... 132
**6. Trouble Shooting Problems ..... 137**
Trouble with Setup ..... 137
Trouble with Building ..... 138
Trouble with Running ..... 138
Tracking Problems by Querying Log Files ..... 139
The coupler log file ..... 139
The cesm log file ..... 139
The CLM log file ..... 142
The DATM log file ..... 142
The batch log files ..... 142
General Advice on Debugging Run time Problems ..... 143
Run in DEBUG mode ..... 143
Run with a smaller set of processors ..... 143
Run in serial mode with a single processor ..... 144
Run at a lower resolution ..... 144
Run a simpler case ..... 144
Run with a debugger ..... 144
**7. Scripts for testing CLM ..... 145**
Testing CLM Using the CESM Test Scripts ..... 145
Testing CLM Using the CLM Specific Stand-Alone Tools Testing Scripts ..... 145
Testing CLM tools using the CLM Stand-Alone Tools Testing Scripts ..... 145
**A. Building the Users-Guide Documentation for CLM ..... 147**

# List of Examples


1-1. Example `user_nl_clm` namelist file ..... **??**
1-2. Default CLM Namelist ..... **??**
1-3. Example `user_nl_clm` namelist adding and removing fields on primary history file ..... **??**
1-4. Example `user_nl_clm` namelist adding auxiliary history files and changing output frequency ..... **??**
1-5. Example `user_nl_clm` namelist removing all history fields ..... **??**
1-6. Example `user_nl_clm` namelist with various ways to average history fields ..... **??**
1-7. Example `user_nl_clm` namelist outputting some files in 1D Vector format ..... 68
2-1. Example of running CLM to create a template file for **interpinic** to interpolate to ..... 89
2-2. Example of building and running **interpinic** to interpolate a 1-degree `finidat` dataset to 10x15 ..89
2-3. Example of running **mksurfdata.pl** to create a 4x5 resolution `fsurdat` for a 1850 simulation year 98
2-4. Getting the raw datasets for **mksurfdata_map** to your local machine using the **check_input_data** script ..... 103
4-1. Example Crop Simulation ..... **??**
4-2. Example Irrigation Simulation ..... **??**
4-3. Example AD_SPINUP Simulation ..... **??**
4-4. Example EXIT_SPINUP Simulation ..... **??**
4-5. Example Final CN Spinup Simulation ..... 115
4-6. Example CNDV Spinup Simulation ..... **??**
4-7. Example Simulation with MOAR Data on yellowstone ..... **??**
4-8. Example Fully Coupled Simulation to Create Data to Force Next Example Simulation ..... **??**
4-9. Example Simulation Forced with Data from the Previous Simulation ..... 118
4-10. Example Transient Simulation with Historical $CO_2$ ..... **??**
5-1. Example of running CLM over a single-point test site in Brazil with the default Qian atmosphere data forcing. ..... 124
5-2. Example of running CLM over the single-point of Mexicocity Mexico with the default Qian atmosphere data forcing. ..... 125
5-3. Example of running CLM over the single-point of Vancouver Canada with supplied atmospheric forcing data for Vancouver. ..... 126
5-4. Example of running **getregional_datasets.pl** to get datasets for a specific region over Alaska ..... 131
5-5. Example of using CLM_USRDAT_NAME to run a simulation using user datasets for a specific region over Alaska ..... 131
5-6. Example of setting up a case with your own atmosphere forcing ..... 133
5-7. Example of DATM streams files with your own forcing for 3-hourly data ..... 134
6-1. Example of cesm_setup problem with missing datasets ..... 137

# Acknowledgments

I want to acknowledge all of the people that helped review or edit the model documentation: David Lawrence, Samuel Levis, Keith Oleson, and Sean Swenson. Thank you for your help in catching errors, and making the document more understandable and readable. Our readers thank you as well, as now it is much easier for them to digest. Any mistakes, or errors are all mine. If you run across one of those errors, please let us know, by following the Section called *Reporting bugs in CLM4* in *Other resources to get help from*. We also want to thank the original authors of PTCLM: Daniel M. Ricciuto, Dali Wang, Peter E. Thornton, Wilfred M. Post, and R. Quinn Thomas for providing a nice addition to the CESM effort (included in previous versions of CESM including CESM1.0.5 but NOT this release unfortunately, but planned for the next release). We also want to thank the folks at University of Michigan Biological Stations (US-UMB) who allowed us to use their Fluxnet station data and import it into our inputdata repository, especially Gil Bohrer the PI on record for this site

# Introduction

The Community Land Model (CLM4 in CESM1.1.1) is the latest in a series of global land models developed by the CESM Land Model Working Group (LMWG) and maintained at the National Center for Atmospheric Research (NCAR). This guide is intended to instruct both the novice and experienced user on running CLM. This guide pertains to the latest version CLM4 in CESM1.1.1 available for download from the public release subversion repository as a part of CESM1.1.1. Documentation may be different if you are using an older version, you should either update to the latest version, or use the documentation inside your own source tree. There is information in the `ChangeLog` file and in the *What is new with CLM4 in CESM1.1.1 since previous public releases?* regarding the changes from previous versions of CESM.

The novice user should read Chapter 1 in detail before beginning work, while the expert user should read *What is new with CLM4 in CESM1.1.1 since previous public releases?* and *Quickstart to using CLM4* chapters, and then use the more detailed chapters as reference. Before novice users go onto more technical problems covered in Chapter 2, Chapter 3, Chapter 4, or Chapter 5 they should know the material covered in Chapter 1 and be able to replicate some of the examples given there.

All users should read the *How to Use This Document* and *Other resources to get help from* sections to understand the document conventions and the various ways of getting help on using CLM4. Users should also read the *What is scientifically validated and functional in CLM4 in CESM1.1.1?* section to see if their planned use of the model is something that has been scientifically validated and well tested. Users that are NOT using NCAR machines or our list of well tested machines should also read the *What are the UNIX utilities required to use CLM?* section to make sure they have all the required UNIX utilities on the system they want to do their work.

Developers that are making changes to CLM either for their own development or for development that they hope will eventually become a part of the main CLM should read the Chapter 7 chapter. We have a suite of test scripts that automatically test many different model configurations and namelist options, as well as ensuring things like restarts are bit-for-bit and the like. It's helpful to use these scripts to ensure your changes are working correctly. And it's far easier to use the automated scripts rather than having to figure out, what to test, how to do it, and then finally do it by hand. If you are using non supported machines you may also want to use the test scripts to make sure your machine is working correctly.

# Introduction to the CLM4 User's Guide

## *What is in here anyway?*

Here in the introduction we first give a simple guide to understand the document conventions in *How to Use This Document*. The next section *What is new with CLM4 in CESM1.1.1 since previous public releases?* gives references to describe the differences between CLM4 in CESM1.1.1 and previous CESM releases both from a scientific as well as a software engineering point of view. For information on previous releases of CLM4 before CLM4 in CESM1.1.0 see the CESM1.0.5 documentation. The next section *Quickstart to using CLM4* is for users that are already experts in using CLM and gives a quickstart guide to the bare details on how to use CLM4. The next *What is scientifically validated and functional in CLM4 in CESM1.1.1?* tells you about what has been extensively tested and scientifically validated (and maybe more importantly) what has NOT. *What are the UNIX utilities required to use CLM?* lists the UNIX utilities required to use CLM4 and is important if you are running on non-NCAR machines, generic local machines, or machines NOT as well tested by us at NCAR. Next we have *Important Notes and Best Practices for Usage of CLM4* to detail some of the best practices for using CLM4 for science. The last introductory section is *Other resources to get help from* which lists different resources for getting help with CESM1.0 and CLM4.

Chapter 1 goes into detail on how to setup and run simulations with CLM4 and especially how to customize cases. Details of **cesm_setup** modes and **build-namelist** options as well as namelist options are given in this chapter.

Chapter 2 gives instructions on the CLM4 tools for creating input datasets for use by CLM, for the expert user. There's an overview of what each tool does, and some general notes on how to build the FORTRAN tools. Then each tool is described in detail along with different ways in which the tool might be used. A final section on how to customize datasets for observational sites for very savvy expert users is given as the last section of this chapter.

As a followup to the tools chapter, Chapter 3 tells how to add files to the XML database for **build-namelist** to use. This is important if you want to use the XML database to automatically select user-created input files that you have created when you setup new cases with CLM.

In Chapter 4, again for the expert user, we give details on how to do some particularly difficult special cases. For example, we give the protocol for spinning up both the CLMCN model and CLM with dynamic vegetation active (CNDV). We give instructions to do a spinup case from a previous case with Coupler history output for atmospheric forcing. We also give instructions on running both the prognostic crop and irrigation models. Lastly we tell the user how to use the DATM model to send historical $CO_2$ data to CLM.

Chapter 5 outlines how to do single-point or regional simulations using CLM4. This is useful to either compare CLM simulations with point observational stations, such as tower sites (which might include your own atmospheric forcing), or to do quick simulations with CLM for example to test a new parameterization. There are several different ways given on how to perform single-point simulations which range from simple PTS_MODE to more complex where you create all your own datasets, tying into Chapter 2 and also Chapter 3 to add the files into the **build-namelist** XML database. The PTCLM python script to run single-point simulations was removed for this release (but is available in the previous CESM1.0.5 release).

The next chapter, Chapter 6 gives some guidance on trouble-shooting problems when using CLM4. It doesn't cover all possible problems with CLM, but gives you some guidelines for things that can be done

for some common problems.

In Chapter 7 we go over the automated testing scripts for validating that the CLM is working correctly. The test scripts run many different configurations and options with CLM making sure that they work, as well as doing automated testing to verify restarts are working correctly, and testing at many different resolutions. In general this is an activity important only for a developer of CLM, but could also be used by users who are doing extensive code modifications and want to ensure that the model continues to work correctly.

In the appendices we talk about some issues that are useful for advanced users and developers of CLM.

Finally in Appendix A we give instructions on how to build the documentation associated with CLM (i.e. how to build this document). This document is included in every CLM distribution and can be built so that you can view a local copy rather than having to go to the CESM website. This also could be useful for developers who need to update the documentation due to changes they have made.

# Important Notes and Best Practices for Usage of CLM4

- When running with CN, it is critical to begin with initial conditions hat are provided with the release or to spin the model up following the CN spinup procedure before conducting scientific runs (see the Section called *Spinning up the biogeochemistry Carbon-Nitrogen Model (CN spinup)* in Chapter 4. Simulations without a proper spinup will effectively be starting from an unvegetated world. See the Section called *Setting Your Initial Conditions File* in Chapter 1 for information on how to provide initial conditions for your simulation.
- Initial condition files are provided for fully coupled BCN and offline ICN cases for 1850 and 2000 at finite volume grids: 1deg (0.9x1.25), 2deg (1.9x2.5), and T31 resolutions. We also have interpolated initial conditions for BCN for 1850 and 2000 for two finite volume grids: 10x15, 4x5 and two HOMME grids (ne30np4 and ne120np4). There's also an initial condition file for ICN with the prognostic crop model for 2000 at 2deg resolution, and one with CLMSP for 2000 at 2deg resolution. We also have initial conditions for offline CNDV for 1850. The 1850 initial condition files are in 'reasonable' equilibrium. The 2000 initial condition files represent the model state for the year 2000, and have been taken from transient simulations. Therefore, by design the year 2000 initial condition files do not represent an equilibrium state. Note also that spinning the 2000 initial conditions out to equilibrium will not reflect the best estimate of the real carbon/nitrogen state for the year 2000.
- Users can generate initial condition files at different resolutions by using the CLM tool **interpinic** to interpolate from one of the provided resolutions to the resolution of interest. Interpolated initial condition files may no longer be in 'reasonable' equilibrium.
- Aerosol deposition is a required field to CLM4 sent from the atmosphere model. Simulations without aerosol deposition will exhibit unreasonably high snow albedos. The model sends aerosol deposition from the atmospheric model (either CAM or DATM). When running with prescribed aerosol the atmosphere model will interpolate the aerosols from 2-degree resolution to the resolution the atmosphere model is running at.

# How to Use This Document

## *Conventions used in the document for code and commands*

This section provides the details in using CLM with the CESM modeling system. Links to descriptions and definitions have been provided in the code below. We use the same conventions used in the CESM documentation as outlined below.

```
Throughout the document this style is used to indicate shell
commands and options, fragments of code, namelist variables, etc.
Where examples from an interactive shell session are presented, lines
starting with > indicate the shell prompt.  A backslash "\" at the end
of a line means the line continues onto the next one (as it does in
standard UNIX shell).  Note that $EDITOR" is used to refer to the
text editor of your choice. $EDITOR is a standard UNIX environment
variable and should be set on most UNIX systems. Comment lines are
signaled with a "#" sign, which is the standard UNIX comment sign as well.
$CSMDATA is used to denote the path to the inputdata directory for
your CESM data.

> This is a shell prompt with commands \
that continues to the following line.
> $EDITOR filename # means you are using a text editor to edit "filename"
# This is a comment line
```

# What is new with CLM4 in CESM1.1.1 since previous public releases?

In this section we list the updates that have occurred to CLM4 since previous public releases of CESM. The CESM1.1.1 Notable Improvements (http://www.cesm.ucar.edu/models/cesm1.1/notable_improvements.html) page gives a synopsis of the changes to all CESM components since the CESM1.0 release. See the CLM section (http://www.cesm.ucar.edu/models/cesm1.1/notable_improvements.html#CLM) for changes particular for CLM. More details are given in the CLM ChangeLog file (http://www.cesm.ucar.edu/models/cesm1.1/clm/models/lnd/clm/doc/ChangeLog).

Previous release pages give similar list of changes for previous versions of the model. The CLM4 in CESM1.0.5 User's Guide (http://www.cesm.ucar.edu/models/cesm1.0/clm/models/lnd/clm/doc/UsersGuide/book1.html) gives information on the updates for versions up to CLM4 in CESM1.0.5.

# Quickstart to using CLM4

Before working with CLM4 read the QuickStart Guide in the CESM1.1.1 Scripts User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm). Once you are familiar with how to setup cases for any type of simulation with CESM you will want to direct your attention to the specifics of using CLM.

For some of the details of setting up cases for CLM4 read the README and text files available from the "models/lnd/clm/doc" directory (see the "CLM Web pages" section for a link to the list of these files). Here are the important ones that you should be familiar with.

1. README (../README) file describing the directory structure.
2. Quickstart.userdatasets (../Quickstart.userdatasets) file describing how to use your own datasets in the model (also see the Section called *Creating your own single-point/regional surface datasets* in Chapter 5).
3. `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file describing known problems in CLM4 (that we expect to eventually fix).
4. `models/lnd/clm/doc/KnownLimitationss` (../KnownLimitations) file describing known limitations in CLM4 and workarounds that we do NOT expect to fix.

The *IMPORTANT_NOTES* file talks about important things for users to know about using the model scientifically. It content is given in the next chapter on "*What is scientifically validated and functional in CLM4 in CESM1.1.1?*".

The *ChangeLog/ChangeSum* talk about advances in different versions of CLM. The content of these files is largely explained in the previous chapter on "*What is new with CLM4 in CESM1.1.1 since previous public releases?*".

Note other directories have README files that explain different components and tools used when running CLM and are useful in understanding how those parts of the model work and should be consulted when using tools in those directories. For more details on configuring and customizing a case with CLM see Chapter 1.

The *Quickstart.GUIDE* (which can be found in `models/lnd/clm/doc)` is repeated here.

```
          Quick-Start to Using cpl7 Scripts for clm4
          ==========================================

Assumptions: You want to use bluefire with clm4
             to do a clm simulation with data atmosphere and the
             latest atm forcing files and settings. You also want to cycle
             the atm data between 1948 to 2004 and you want to run at
             1.9x2.5 degree resolution.

Process:

   # Create the case

   cd scripts

   ./create_newcase -case <testcase> -mach bluefire_ibm -res f19_g16 -compset I4804
   (./create_newcase -help -- to get help on the script)
TRUE

   # Setup the case

   cd <testcase>
   ./xmlchange id1=val1,id2=val2  # to make changes to any settings in the env_*.xml files
```

```
   ./cesm_setup
   (./cesm_setup -help -- to get help on the script, this creates the ./<testcase>.run  \
   script)

   # Add any namelist changes to the user_nl_* files

   $EDITOR user_nl_*

   # Compile the code

   ./<testcase>.build

   # Submit the run

   ./<testcase>.submit

Information on Compsets:

     "I" compsets are the ones with clm and datm7 without ice and ocean.
     Most of the "I" compsets use the CLM_QIAN data with solar following
     the cosine of solar zenith angle, precipitation constant, and other
     variables linear interpolated in time (and with appropriate time-stamps on
     the date). Some of the I compsets are:

     Name    (short-name): Description
     ------------------------------------------------------------------------
     I_2000                (I):             CLM to simulate year=2000
     I_2000_1PTFRC         (I1PT):          CLM to simulate year=2000 with single-point forcing
     I_2000_CN             (ICN):           CLM to simulate year=2000 with Carbon-Nitrogen BGC  \
   model (CN)
     I_1850                (I1850):         CLM to simulate year=1850
     I_1850_CN             (I1850CN):       CLM to simulate year=1850 with Carbon-Nitrogen BGC  \
   model (CN)
     I_1850_SPINUP_3HrWx_CN (I1850SPINUPCN): CLM to simulate year=1850 with MOAR forcing to spinup  \
   Carbon-Nitrogen BGC model (CN)
     I_1948_2004           (I4804):         CLM running with atm data over 1948-2004
     I_1850-2000           (I8520):         CLM with transient PFT over 1850-2000
     I_1948-2004_CN        (I4804CN):       CLM with CN on running with atm data over 1948-2004
     I_1850-2000_CN        (I20TRCN):       CLM with CN on with transient PFT over 1850-2000
     I_RCP2.6_CN           (IRCP26CN):      CLM with CN on with transient PFT over 1850-2100 for  \
   RCP=2.6 scenario
     I_RCP4.5_CN           (IRCP45CN):      CLM with CN on with transient PFT over 1850-2100 for  \
   RCP=4.5 scenario
     I_RCP6.0_CN           (IRCP60CN):      CLM with CN on with transient PFT over 1850-2100 for  \
   RCP=6.0 scenario
     I_RCP8.5_CN           (IRCP85CN):      CLM with CN on with transient PFT over 1850-2100 for  \
   RCP=8.5 scenario

Automatically resubmitting jobs:

   After doing a short simulation that you believe is correct

   ./xmlchange CONTINUE_RUN=TRUE

   # Change RESUBMIT to number greater than 0, and CONTINUE_RUN to TRUE...

   ./<testcase>.submit
```

# What is scientifically validated and functional in CLM4 in CESM1.1.1?

In this section we go over what has been extensively tested and scientifically validated with CLM4, and maybe more importantly what has NOT been tested and may NOT be scientifically validated. You can use all features of CLM, but need to realize that some things haven't been tested extensively or validated scientifically. When you use these features you may run into trouble doing so, and will need to do your own work to make sure the science is reasonable.

## Standard Configuration and Namelist Options that are Validated

In this release "NO" standard configuration are scientifically validated. Changes to the science of the model are minimul since CESM1.0.5 so we expect answers to be very similar to using it.

In the sections below we go through configuration and/or namelist options or modes that the user should be especially wary of using. You are of course free to use these options, and you may find that they work functionally. Although in some cases you will find issues even with functionality of using them. If so you will need to test, debug and find solutions for these issues on your own. But in every case you will need to go through more extensive work to validate these options from a scientific standpoint.

## Configure Modes NOT scientifically validated, documented, supported or, in some cases, even advised to be used:

1. `C13`

   `(-c13)`

   The C13 mode for bgc=cn is NOT scientifically validated or documented and is NOT recommended for use. It is also deprecated and will be removed in the next release (although CLM4.5 will have namelist options for both C13 and C14).

2. `SNICAR_FRC`

   `(-snicar_frc)`

   This mode is tested and functional, but is NOT constantly scientifically validated, and should be considered experimental.

# Namelist options that should NOT be exercised:

## Build-Namelist options that should NOT be exercised:

1. *-irrig with -bgc cn* We have only run the irrigation model with CLMSP (i.e. without the CN model). We recommend that if you want to run the irrigation model with CN, that you do a spinup. But, more than that you may need to make adjustments to `irrig_factor` in `models/lnd/clm/src/biogeophys/CanopyFluxesMod.F90`. See the notes on this in the description of the irrigation model in the Technical Descriptions of the Interactive Crop Management and Interactive Irrigation Models (http://www.cesm.ucar.edu/models/cesm1.1//CLMcropANDirrigTechDescriptions.pdf).
2. *-irrig with -crop on* Irrigation doesn't work with the prognostic crop model. Irrigation is only applied to generic crop currently, which negates it's practical usage. We also have a known problem when both are on (see bug 1326 in the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file). If you try to run in this mode, the CLM **build-namelist** will return with an error.
3. *-lnd_res:* Fine-mesh mode, functional, but experimental
4. *-rcp:* Representative Concentration Pathway (RCP) for future scenarios, functional for limited resolutions, but experimental
5. *-datm_*:* All options that start with "datm_" they are only used for CLM stand-alone testing.
6. *-drv_*:* All options that start with "drv_" they are only used for CLM stand-alone testing.

## Namelist items that should NOT be exercised:

1. *suplnitro='ALL'* The suplnitro namelist option to the CN Biogeochemistry model supplies unlimited nitrogen and therefore vegetation is over-productive in this mode.
2. *urban_traffic:* Not currently functional

# What are the UNIX utilities required to use CLM?

Running the CLM requires a suite of UNIX utilities and programs and you should make sure you have all of these available before trying to go forward with using it. If you are missing one of these you should contact the systems administrator for the machine you wish to run on and make sure they are installed.

List of utilities required for CESM in the "CESM1.1.1 Software/Operating System Prerequisites" section in http://www.ce
UNIX bash shell (for some of the CLM tools scripts)
NCL (for some of the offline tools for creating/modifying CLM input datasets see Chapter 2 for more information on NCL
Python (optional, needed for PTCLM)
xsltproc, docbook and docbook utilities (optional, needed to build the Users-Guide)
protex and latex2html (optional, needed to build the Code-Reference Guide)

# Other resources to get help from

In addition to this users-guide there are several other resources that are available to help you use CLM4. The first one is the CESM1.1.1 User's-Guide, which documents the entire process of creating cases with CESM1.1.1. The next is the CESM bulletin board which is a web-site for exchanging information between users of CESM. There are also CLM web-pages specific for CLM, and finally there is an email address to report bugs that you find in CESM1.1.1.

## The CESM User's-Guide

CLM4 in CESM1.1.1 is always run from within the standard CESM1.1.1 build and run scripts. Therefore, the user of CLM4 should familiarize themselves with the CESM1.1.1 scripts and understand how to work with them. User's-Guide documentation on the CESM1.1.1 scripts are available from the following web-page. The purpose of this CLM4 in CESM1.1.1 User's Guide is to give the CLM4 user more complete details on how to work with CLM and the set of tools that support CLM, as well as to give examples that are unique to the use of CLM. However, the CESM1.1.1 Scripts User's-Guide remains the primary source to get detailed information on how to build and run the CESM system.

cesmrel; Scripts User's-Guide (http://www.cesm.ucar.edu/models/cesm1.1//cesm/doc/usersguide/book1.html) (http://www

## The CESM Bulletin Board

There is a rich and diverse set of people that use the CESM, and often it is useful to be in contact with others to get help in solving problems or trying something new. To facilitate this we have an online Bulletin Board for questions on the CESM. There are also different sections in the Bulletin Board for the different component models or for different topics.

CESM Online Bulletin Board (http://bb.cgd.ucar.edu/)

## The CLM web pages

The main CLM web page contains information on the CLM, it's history, developers, as well as downloads for previous model versions. There are also documentation text files in the models/lnd/clm/doc directory that give some quick information on using CLM.

CLM web page (http://www.cgd.ucar.edu/tss/clm/)
CLM Documentation Text Files (../)

Also note that several of the XML database files can be viewed in a web browser to get a nice table of namelist options, namelist defaults, or compsets. Simply view them as a local file and bring up one of the following files:
models/lnd/clm/bld/namelist_files/namelist_definition.xml (../../bld/namelist_files/namelist_definition.xml) -- definition of
models/lnd/clm/bld/namelist_files/namelist_defaults_clm.xml (../../bld/namelist_files/namelist_defaults_clm.xml) -- defau

scripts/ccsm_utils/Case.template/config_definition.xml (../../../../../scripts/ccsm_utils/Case.template/config_definition.xml)
scripts/ccsm_utils/Case.template/config_compsets.xml (../../../../../scripts/ccsm_utils/Case.template/config_compsets.xml)
models/lnd/clm/bld/namelist_files/history_fields.xml (../../bld/namelist_files/history_fields.xml) -- definition of CLM histo

# Reporting bugs in CLM4

If you have any problems, additional questions, bug reports, or any other feedback, please send an email to <cesmhelp@cgd.ucar.edu>. If you find bad, wrong, or misleading information in this users guide send an email to <erik@ucar.edu>. The current list of known issues for CLM4 in CESM1.1.1 is in the models/lnd/clm/doc/KnownBugs (../KnownBugs) file, and the list of issues for CESM1.1.1 is at... http://www.cesm.ucar.edu/models/cesm1.1//tags/cesm1_1_1/#PROBLEMS (http://www.cesm.ucar.edu/models/cesm1.1//tags/cesm1_1_1/#PROBLEMS).

# Some Acronymn's We'll be Using

CAM

: Community Atmosphere Model (CAM)

CESM

: Community Earth System Model (CESM)

CLM

: Community Land Model (CLM)

CLMCN

: Community Land Model (CLM) with Carbon Nitrogen (CN) Biogeochemistry

CLMSP

: Community Land Model (CLM) with Satellite Phenology (SP)

CLMU

: Community Land Model (CLM) Urban Model

DATM

: Data Atmosphere Model (DATM)

ESMF

: Earth System Modelling Framework

NCAR

: National Center for Atmospheric Research

PTCLM

: PoinT CLM

SCRIP

: Spherical Coordinate Remapping and Interpolation Package (SCRIP)

# Chapter 1. How to customize the configuration for a case with CLM

The CESM1.1.1 User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm) gives you the details on how to create, setup, build, and run a case. That is the document to give you the details on using the CESM scripts. The purpose of this document is to give you the details when using CESM with CLM on how to customize and use advanced features in CLM. You should be familiar with the CESM1.1.1 User's Guide and how to setup cases with CESM1.1.1 before referring to this document.

In this chapter we deal with three different ways of customizing a case: Choosing a compset, Customizing Configuration options, and customizing the CLM Namelist. There are many different compsets that use CLM and many are setup to enable special features of CLM from the start. So the first thing you want to be familiar with are the different options in the compsets. The next section shows the different options for customizing the configuration options for CLM. Here we introduce the CLM **configure** and **build-namelist** scripts and how using the options in `env_build.xml` and `env_run.xml` you can customize the configuration and the namelist.

## Choosing a compset using CLM

When setting up a new case one of the first choices to make is which "component set" (or compset) to use. The compset refers to which component models are used as well as specific settings for them. We label the different types of compsets with a different letter of the alphabet from "A" (for all data model) to "X" (for all dead model). The compsets of interest when working with CLM are the "I" compsets (which contain CLM with a data atmosphere model and a stub ocean, and stub sea-ice models), "E" and "F" compsets (which contain CLM with the active atmosphere model (CAM), prescribed sea-ice model, and a data ocean model), and "B" compsets which have all active components. Below we go into details on the "I" compsets which emphasize CLM as the only active model, and just mention the two other categories.

When working with CLM you usually want to start with a relevant "I" compset before moving to the more complex cases that involve other active model components. The "I" compsets can exercise CLM in a way that is similar to the coupled modes, but with much lower computational cost and faster turnaround times.

### Compsets coupled to data atmosphere and stub ocean/sea-ice ("I" compsets)

1. `I_2000(I)` CLM with QIAN atm input data for 1972-2004 and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000
2. `I_1850(I1850)` CLM with QIAN atm input data for 1948 to 1972 and Satellite phenology (SP), CO2 level and Aerosol deposition for 1850
3. `I_1850_CN(I1850CN)` CLM with QIAN atm input data for 1948 to 1972 and CN (Carbon Nitrogen) biogeochemistry, 1850 CO2 and Aerosol deposition

4. I_1850_SPINUP_3HrWx_CN(I1850SPINUPCN) CLM with BCN CPLHIST 3-hourly weather forcing data and half-hourly solar for 1850 spinup of CN (Carbon Nitrogen) biogeochemistry, CO2 level and Aerosol deposition for 1850
5. I_2000_1PTFRC(I1PT) CLM with single point tower site atm input data and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000
6. I_1850-2000(I20TR) CLM with QIAN atm input data for 1948 to 1972 and transient Satellite phenology (SP), and Aerosol deposition from 1850 to 2000 and 2000 CO2 level
7. I_1850-2000_CN(I20TRCN) CLM with QIAN atm input data for 1948 to 1972 and transient CN, Aerosol dep from 1850 to 2000 and 2000 CO2 level
8. I_1948-2004(I4804) CLM with QIAN atm input data for 1948 to 2004 and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000
9. I_1948-2004_CN(I4804CN) CLM with QIAN atm data for 1948 to 2004 and CN (Carbon Nitrogen) biogeochemistry, CO2 level and Aerosol deposition for 2000
10. I_2000_CN(ICN) CLM with QIAN atm input data for 1972-2004 and CN (Carbon Nitrogen) biogeochemistry, CO2 level and Aerosol deposition for 2000
11. I_TEST_2003_CN(ICNTEST) Test active land model with QIAN atm input data for just 2002-2003 and CN (Carbon-Nitrogen), CO2 level and Aerosol deposition for 2000 (BECAUSE OF THE SHORT FORCING PERIOD -- DO NOT USE FOR LONG SIMULATIONS)
12. I_2000_GLC(IG) Active glacier model and active land model with QIAN atm input data for 1972-2004 and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000
13. I_1850_GLC(IG1850) CLM and CISM with QIAN atm input data for 1948 to 1972 and Satellite phenology (SP), 1850 CO2 and Aerosol deposition
14. I_1850-2000_GLC(IG20TR) CLM and CISM with QIAN atm data for 1948 to 1972 and transient Satellite phenology (SP), and Aerosol deposition from 1850 to 2000 and 2000 CO2 level
15. I_1850-2000_CN_GLC(IG20TRCN) CLM and CISM with QIAN atm input data for 1948 to 1972 and transient CN, Aerosol dep from 1850 to 2000 and 2000 CO2 level
16. I_1948-2004_GLC(IG4804) CLM and CISM with QIAN atm input data for 1948 to 2004 and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000
17. I_1948-2004_CN_GLC(IG4804CN) CLM and CISM with QIAN atm data for 1948 to 2004 and CN (Carbon Nitrogen) biogeochemistry, CO2 level and Aerosol deposition for 2000
18. I_2000_CN_GLC(IGCN) CLM and CISM with QIAN input data for 1972-2004 and CN (Carbon Nitrogen) biogeochemistry, CO2 level and Aerosol deposition for 2000
19. I_RCP2.6_CN_GLC(IGRCP26CN) CLM and glacier model, RCP2.6 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
20. I_RCP4.5_CN_GLC(IGRCP45CN) CLM and glacier model, RCP4.5 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
21. I_RCP6.0_CN_GLC(IGRCP60CN) CLM and glacier model, RCP6.0 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
22. I_RCP8.5_CN_GLC(IGRCP85CN) CLM and glacier model, RCP8.5 future scenario, with CN in CLM, QIAN atm input data for 1972-2004, 2000 CO2 level

23. `I_RCP2.6_CN`(`IRCP26CN`) CLM, RCP2.6 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
24. `I_RCP4.5_CN`(`IRCP45CN`) CLM, RCP4.5 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
25. `I_RCP6.0_CN`(`IRCP60CN`) CLM, RCP6.0 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
26. `I_RCP8.5_CN`(`IRCP85CN`) CLM, RCP8.5 future scenario, with CN in CLM, QIAN atm input data for 1972 to 2004, 2000 CO2 level
27. `I_TEST_2003`(`ITEST`) Test active land model with QIAN atm input data for just 2002-2003 and Satellite phenology (SP), CO2 level and Aerosol deposition for 2000 (BECAUSE OF THE SHORT FORCING PERIOD -- DO NOT USE FOR LONG SIMULATIONS)

### Compsets coupled to active atmosphere with data ocean

CAM compsets are compsets that start with "E" or "F" in the name. They are described more fully in the scripts documentation or the CAM documentation. "E" compsets have a slab ocean model while "F" compsets have a data ocean model.

### Fully coupled compsets with fully active ocean, sea-ice, and atmosphere

Fully coupled compsets are compsets that start with "B" in the name. They are described more fully in the scripts documentation.

### Conclusion to choosing a compset

We've introduced the basic type of compsets that use CLM and given some further details for the "standalone CLM" (or "I" compsets). The config_compsets.xml (../../../../../scripts/ccsm_utils/Case.template/config_compsets.xml) lists all of the compsets and gives a full description of each of them. In the next section we look into customizing the setup time options for compsets using CLM.

## Customizing the CLM setup

The "Creating a Case" section of the CESM1.1.1 Scripts User's-Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm) gives instructions on creating a case. What is of interest here is how to customize your use of CLM for the case that you created.

For CLM when **preview_namelist**, **$CASE.build**, or **$CASE.run** are called there are two steps that take place:

1. The CLM "**configure**" script is called to setup the build-time configuration for CLM (more information on **configure** is given in the Section called *More information on the CLM **configure** script*). The env variables for **configure** are locked after the **$CASE.build** step. So the results of the CLM **configure** are locked after the build has taken place.
2. The CLM "**build-namelist**" script is called to generate the run-time namelist for CLM (more information on **build-namelist** is given below in the Section called *Definition of Namelist items and their default values*.

When customizing your case at the **cesm_setup** step you are able to modify the process by effecting either one or both of these steps. The CLM "**configure**" and "**build-namelist**" scripts are both available in the "models/lnd/clm/bld" directory in the distribution. Both of these scripts have a "-help" option that is useful to examine to see what types of options you can give either of them.

There are five different types of customization for the configuration that we will discuss: CLM4 in CESM1.1.1 build-time options, CLM4 in CESM1.1.1 run-time options, User Namelist, other noteworthy CESM1.1.1 configuration items, the CLM **configure** script options, and the CLM **build-namelist** script options.

Information on all of the script, configuration, build and run items is found under `scripts/ccsm_utils/Case.template` in the `config_definition.xml` (../../../../../scripts/ccsm_utils/Case.template/config_definition.xml) file.

## CLM Script configuration items

Below we list each of the CESM configuration items that are specific to CLM. All of these are available in your: `env_build.xml` and `env_run.xml` files.

CLM_CONFIG_OPTS
CLM_BLDNML_OPTS
CLM_NAMELIST_OPTS
CLM_FORCE_COLDSTART
CLM_NML_USE_CASE
CLM_USRDAT_NAME
CLM_CO2_TYPE

For the precedence of the different options to **build-namelist** see the section on precedence below.

The first item CLM_CONFIG_OPTS has to do with customizing the CLM build-time options for your case, the rest all have to do with generating the namelist.

CLM_CONFIG_OPTS

The option CLM_CONFIG_OPTS is all about passing command line arguments to the CLM **configure** script. It is important to note that some compsets, may already put a value into the CLM_CONFIG_OPTS variable. You can still add more options to your CLM_CONFIG_OPTS but make sure you add to what is already there rather than replacing it. Hence, we recommend using the "-append" option to the xmlchange script. In the Section called *More information on the CLM **configure** script* below we will go into more details on options that can be customized in the CLM "**configure**" script. It's also important to note that the **clm.buildnml.csh** script may already invoke certain CLM **configure** options and as such those command line options are NOT going to be available to change at this step (nor would you want to change them). The options to CLM **configure** are given with the "-help" option which is given in the Section called *More information*

*on the CLM **configure** script*.

> **Note:** CLM_CONFIG_OPTS is locked after the **$CASE.build** script is run. If you want to change something in CLM_CONFIG_OPTS you'll need to clean the build and rerun **$CASE.build**. The other env variables can be changed at run-time so are never locked.

CLM_NML_USE_CASE

CLM_NML_USE_CASE is used to set a particular set of conditions that set multiple namelist items, all centering around a particular usage of the model. To list the valid options do the following:

```
> cd models/lnd/clm/doc
> ../bld/build-namelist -use_case list
```

The output of the above command is:

```
build-namelist - use cases: 1850-2100_rcp2.6_glacierMEC_transient 1850-2100_rcp2.6_transient  \
   1850-2100_rcp4.5_glacierMEC_transient 1850-2100_rcp4.5_transient  \
   1850-2100_rcp6_glacierMEC_transient 1850-2100_rcp6_transient  \
   1850-2100_rcp8.5_glacierMEC_transient 1850-2100_rcp8.5_transient 1850_control  \
   1850_glacierMEC_control 2000-2100_rcp8.5_transient 2000_control 2000_glacierMEC_control  \
   20thC_glacierMEC_transient 20thC_transient glacierMEC_pd pergro0_pd pergro_pd stdurbpt_pd
Use cases are:...

1850-2100_rcp2.6_glacierMEC_transient = Simulate transient land-use, and aerosol deposition changes  \
   with historical data from 1850 to 2005 and then with the RCP2.6 scenario from IMAGE

1850-2100_rcp2.6_transient = Simulate transient land-use, and aerosol deposition changes with  \
   historical data from 1850 to 2005 and then with the RCP2.6 scenario from IMAGE

1850-2100_rcp4.5_glacierMEC_transient = Simulate transient land-use, and aerosol deposition changes  \
   with historical data from 1850 to 2005 and then with the RCP4.5 scenario from MINICAM

1850-2100_rcp4.5_transient = Simulate transient land-use, and aerosol deposition changes with  \
   historical data from 1850 to 2005 and then with the RCP4.5 scenario from MINICAM

1850-2100_rcp6_glacierMEC_transient = Simulate transient land-use, and aerosol deposition changes  \
   with historical data from 1850 to 2005 and then with the RCP6 scenario from AIM

1850-2100_rcp6_transient = Simulate transient land-use, and aerosol deposition changes with  \
   historical data from 1850 to 2005 and then with the RCP6 scenario from AIM

1850-2100_rcp8.5_glacierMEC_transient = Simulate transient land-use, and aerosol deposition changes  \
   with historical data from 1850 to 2005 and then with the RCP8.5 scenario from MESSAGE

1850-2100_rcp8.5_transient = Simulate transient land-use, and aerosol deposition changes with  \
   historical data from 1850 to 2005 and then with the RCP8.5 scenario from MESSAGE

   1850_control = Conditions to simulate 1850 land-use
1850_glacierMEC_control = Running an IG case for 1850 conditions with the ice sheet model glimmer
2000-2100_rcp8.5_transient = Simulate transient land-use, and aerosol deposition changes with  \
   historical data from 2000 to 2005 and then with the RCP8.5 scenario from MESSAGE

   2000_control = Conditions to simulate 2000 land-use
2000_glacierMEC_control = Running an IG case for 2000 conditions with the ice sheet model glimmer
20thC_glacierMEC_transient = Simulate transient land-use, and aerosol deposition changes from 1850  \
   to 2005
20thC_transient = Simulate transient land-use, and aerosol deposition changes from 1850 to 2005
  glacierMEC_pd = Running an IG case with the ice sheet model glimmer
     pergro0_pd = Perturbation error growth test with unperturbed initial conditions
      pergro_pd = Perturbation error growth test with initial conditions perturbed by roundoff  \
   level
    stdurbpt_pd = Standard Urban Point Namelist Settings
```

**Note:** See the the Section called *Precedence of Options* section for the precedence of this option relative to the others.

CLM_BLDNML_OPTS

The option CLM_BLDNML_OPTS is for passing options to the CLM "**build-namelist**" script. As with the CLM "**configure**" script the CLM **clm.buildnml.csh** may already invoke certain options and as such those options will NOT be available to be set here. The best way to see what options can be sent to the "**build-namelist**" script is to do

```
> cd models/lnd/clm/bld
> ./build-namelist -help
```

Here is the output from the above.

```
./SYNOPSIS
     build-namelist [options]

     Create the namelist for CLM
OPTIONS
     -[no-]chk_res            Also check [do NOT check] to make sure the resolution and
                              land-mask is valid.
     -clm_demand "list"       List of variables to require on clm namelist besides the usuals.
                              "-clm_demand list" to list valid options.
                              (can include a list member "null" which does nothing)
     -clm_startfile "file"    CLM restart file to start from.
     -clm_start_type "type"   Start type of simulation
                              (default, cold, arb_ic, startup, continue, or branch)
                              (default=do the default type for this configuration)
                              (cold=always start with arbitrary initial conditions)
                              (arb_ic=start with arbitrary initial conditions if
                               initial conditions don't exist)
                              (startup=ensure that initial conditions are being used)
     -clm_usr_name     "name" Dataset resolution/descriptor for personal datasets.
                              Default: not used
                              Example: 1x1pt_boulderCO_c090722 to describe location,
                                       number of pts, and date files created
     -co2_type "value"        Set CO2 the type of CO2 variation to use.
     -co2_ppmv "value"        Set CO2 concentration to use when co2_type is constant (ppmv).
     -config "filepath"       Read the given CLM configuration cache file.
                              Default: "config_cache.xml".
     -csmdata "dir"           Root directory of CESM input data.
                              Can also be set by using the CSMDATA environment variable.
     -d "directory"           Directory where output namelist file will be written
                              Default: current working directory.
     -drydep                  Also produce a drydep_inparm namelist that will go into the
                              "drv_flds_in" file for the driver to pass dry-deposition
                              to the atm.
                              (Note: buildnml.csh copies the file for use by the driver)
     -glc_grid "grid"         Glacier model grid and resolution when glacier model,
                              Only used if glc_nec > 0 for determining fglcmask
                              Default:  gland5UM
                              (i.e. gland20, gland10 etcetera)
     -glc_nec <name>          Glacier number of elevation classes [0 | 3 | 5 | 10]
                              (default is 0) (standard option with land-ice model is 10)
     -glc_smb <value>         Only used if glc_nec > 0
                              If .true., pass surface mass balance info to GLC
                              If .false., pass positive-degree-day info to GLC
                              Default: true
     -help [or -h]            Print usage to STDOUT.
     -ignore_ic_date          Ignore the date on the initial condition files
                              when determining what input initial condition file to use.
     -ignore_ic_year          Ignore just the year part of the date on the initial condition files
 \
                                 when determining what input initial condition file to use.
     -infile "filepath"       Specify a file (or list of files) containing namelists to
                              read values from.

                              If used with a CLM build with multiple ensembles (ninst_lnd>1)
                              and the filename entered is a directory to files of the
```

```
                           form filepath/filepath and filepath/filepath_$n where $n
                           is the ensemble member number. the "filepath/filepath"
                           input namelist file is the master input namelist file
                           that is applied to ALL ensemble members.

                           (by default for CESM this is setup for files of the
                            form $CASEDIR/user_nl_clm/user_nl_clm_????)
    -inputdata "filepath"  Writes out a list containing pathnames for required input datasets in
\
                              file specified.
    -irrig                 Seek surface datasets with irrigation turned on.
    -l_ncpl "LND_NCPL"     Number of CLM coupling time-steps in a day.
    -lnd_frac "domainfile" Land fraction file (the input domain file)
    -mask "landmask"       Type of land-mask (default, navy, gx3v5, gxlv5 etc.)
                           "-mask list" to list valid land masks.
    -megan                 Also produce a megan_emis_nl namelist that will go into the
                           "drv_flds_in" file for the driver to pass VOC's to atm.
                           MEGAN (Model of Emissions of Gases and Aerosols from Nature)
                           (Note: buildnml.csh copies the file for use by the driver)
    -namelist "namelist"   Specify namelist settings directly on the commandline by supplying
                           a string containing FORTRAN namelist syntax, e.g.,
                              -namelist "&clm_inparm dt=1800 /"
    -[no-]note             Add note to output namelist  [do NOT add note] about the
                           arguments to build-namelist.
    -rcp "value"           Representative concentration pathway (rcp) to use for
                           future scenarios.
                           "-rcp list" to list valid rcp settings.
    -res "resolution"      Specify horizontal grid.  Use nlatxnlon for spectral grids;
                           dlatxdlon for fv grids (dlat and dlon are the grid cell size
           in degrees for latitude and longitude respectively)
                           "-res list" to list valid resolutions.
    -s                     Turns on silent mode - only fatal messages issued.
    -sim_year "year"       Year to simulate for input datasets
                           (i.e. 1850, 2000, 1850-2000, 1850-2100)
                           "-sim_year list" to list valid simulation years
    -test                  Enable checking that input datasets exist on local filesystem.
    -verbose [or -v]       Turn on verbose echoing of informational messages.
    -use_case "case"       Specify a use case which will provide default values.
                           "-use_case list" to list valid use-cases.
    -version               Echo the SVN tag name used to check out this CLM distribution.


Note: The precedence for setting the values of namelist variables is (highest to lowest):
      0. namelist values set by specific command-line options, like, -d, -sim_year
            (i.e.  CLM_BLDNML_OPTS env_run variable)
      1. values set on the command-line using the -namelist option,
            (i.e. CLM_NAMELIST_OPTS env_run variable)
      2. values read from the file(s) specified by -infile,
            (i.e.  user_nl_clm files)
      3. datasets from the -clm_usr_name option,
            (i.e.  CLM_USRDAT_NAME env_run variable)
      4. values set from a use-case scenario, e.g., -use_case
            (i.e.  CLM_NML_USE_CASE env_run variable)
      5. values from the namelist defaults file.
```

The **clm.buildnml.csh** script already sets the resolution and mask as well as the CLM **configure** file, and defines an input namelist and namelist input file, and the output namelist directory, and sets the start-type (from RUN_TYPE), namelist options (from CLM_NAMELIST_OPTS), co2_ppmv (from CCSM_CO2_PPMV, co2_type (from CLM_CO2_TYPE), lnd_frac (from LND_DOMAIN_PATH and LND_DOMAIN_FILE), l_ncpl (from LND_NCPL, glc_grid, glc_smb, glc_nec (from GLC_GRID, GLC_SMB, and GLC_NEC), and "clm_usr_name" is set (to CLM_USRDAT_NAME >when the grid is set to CLM_USRDAT_NAME. Hence only the following different options can be set:

1. -chk_res
2. -clm_demand
3. -drydep
4. -ignore_ic_date

5. -ignore_ic_year
6. -irrig
7. -megan
8. -note
9. -rcp
10. -sim_year
11. -verbose

"-chk_res" ensures that the resolution chosen is supported by CLM. If the resolution is NOT supported it will cause the CLM **build-namelist** to abort when run. So when either **preview_namelist**, **$CASE.build** or **$CASE.run** is executed it will abort early. Since, the CESM scripts only support certain resolutions anyway, in general this option is NOT needed in the context of running CESM cases.

"-clm_demand" asks the **build-namelist** step to require that the list of variables entered be set. Typically, this is used to require that optional filenames be used and ensure they are set before continuing. For example, you may want to require that `fpftdyn` be set to get dynamically changing vegetation types. To do this you would do the following.

```
> ./xmlchange CLM_BLDNML_OPTS="-clm_demand fpftdyn"
```

To see a list of valid variables that you could set do this:

```
> cd models/lnd/clm/doc
> ../bld/build-namelist -clm_demand list
```

> **Note:** Using a 20th-Century transient compset or the 20thC_transient use-case using CLM_NML_USE_CASE would set this as well, but would also use dynamic nitrogen and aerosol deposition files, so using -clm_demand would be a way to get *just* dynamic vegetation types and NOT the other files as well.

"-drydep" adds the dry-deposition namelist to the driver. This is a driver namelist, but adding the option here has CLM **build-namelist** create the `drv_flds_in` file that the driver will copy over and use.

"-ignore_ic_date" ignores the Initial Conditions (IC) date completely for finding initial condition files to startup from. Without this option or the "-ignore_ic_year" option below, the date of the file comes into play.

"-ignore_ic_year" ignores the Initial Conditions (IC) year for finding initial condition files to startup from. The date is used, but the year is ignored. Without this option or the "-ignore_ic_date" option below, the date and year of the file comes into play.

When "-irrig" is used **build-namelist** will try to find surface datasets that have the irrigation model enabled.

"-megan" adds the MEGAN model Biogenic Volatile Organic Componds (BVOC) namelist to the driver. This is a driver namelist, but adding the option here has CLM **build-namelist** create the `drv_flds_in` file that the driver will copy over and use.

"-note" adds a note to the bottom of the namelist file, that gives the details of how **build-namelist** was called, giving the specific command-line options given to it.

"-rcp" is used to set the representative concentration pathway for the future scenarios you want the data-sets to simulate conditions for, in the input datasets. To list the valid options do the following:

```
> cd models/lnd/clm/doc
> ../bld/build-namelist -rcp list
```

"-sim_year" is used to set the simulation year you want the data-sets to simulate conditions for in the input datasets. The simulation "year" can also be a range of years in order to do simulations with changes in the dataset values as the simulation progresses. To list the valid options do the following:

```
> cd models/lnd/clm/doc
> ../bld/build-namelist -sim_year list
```

CLM_NAMELIST_OPTS

The option CLM_NAMELIST_OPTS is for passing namelist items into one of the CLM namelists.

**Important:** For character namelist items you need to use "'" as quotes for strings so that the scripts don't get confused with other quotes they use.

Example, you want to set `hist_dov2xy` to `.false.` so that you get vector output to your history files. To do so edit `env_run.xml` and add a setting for `hist_dov2xy`. So do the following:

```
> ./xmlchange CLM_NAMELIST_OPTS="hist_dov2xy=.false."
```

Example, you want to set `hist_fincl1` to add the variable `'HK'` to your history files. To do so edit `env_run.xml` and add a setting for `hist_fincl1`. So do the following:

```
> ./xmlchange CLM_NAMELIST_OPTS="hist_fincl1=&amp;apos;HK&amp;apos;"
```

For a list of the history fields available see CLM History Fields (../../bld/namelist_files/history_fields.xml).

**Note:** See the the Section called *Precedence of Options* section for the precedence of this option relative to the others.

CLM_FORCE_COLDSTART

CLM_FORCE_COLDSTART when set to `on`, *requires* that your simulation do a cold start from arbitrary initial conditions. If this is NOT set, it will use an initial condition file if it can find an appropriate one, and otherwise do a cold start. CLM_FORCE_COLDSTART is a good way to ensure that you are doing a cold start if that is what you want to do.

CLM_USRDAT_NAME

CLM_USRDAT_NAME provides a way to enter your own datasets into the namelist. The files you create must be named with specific naming conventions outlined in: the Section called *Creating your own single-point/regional surface datasets* in Chapter 5. To see what the expected names of the files are, use the **queryDefaultNamelist.pl** to see what the names will need to be. For example if your CLM_USRDAT_NAME will be "1x1_boulderCO", with a "navy" land-mask, constant simulation year range, for 1850, the following will list what your filenames should be:

```
> cd models/lnd/clm/bld
> queryDefaultNamelist.pl -usrname "1x1_boulderCO" -options \
mask=navy,sim_year=1850,sim_year_range="constant"  -csmdata $CSMDATA
```

An example of using CLM_USRDAT_NAME for a simulation is given in Example 5-5.

**Note:** See the the Section called *Precedence of Options* section for the precedence of this option relative to the others.

CLM_CO2_TYPE

CLM_CO2_TYPE sets the type of input $CO_2$ for either "constant", "diagnostic" or prognostic". If "constant" the value from CCSM_CO2_PPMV will be used. If "diagnostic" or "prognostic" the values MUST be sent from the atmosphere model. For more information on how to send $CO_2$ from the data atmosphere model see the Section called *Running stand-alone CLM with transient historical $CO_2$ concentration* in Chapter 4.

## User Namelist

CLM_NAMELIST_OPTS as described above allows you to set any extra namelist items you would like to appear in your namelist. However, it only allows you a single line to enter namelist items, and strings must be quoted with ' which is a bit awkward. If you have a long list of namelist items you want to set (such as a long list of history fields) a convenient way to do it is to add to the `user_nl_clm` that is created after the **cesm_setup** command runs. The file needs to be in valid FORTRAN namelist format (with the exception that the namelist name &namelist and the end of namelist marker "/" are excluded". The **preview_namelist** or **$CASE.run** step will abort if there are syntax errors. All the variable names must be valid and the values must be valid for the datatype and any restrictions for valid values for that variable. Here's an example `user_nl_clm` namelist that sets a bunch of history file related items, to create output history files monthly, daily, every six and 1 hours.

**Example 1-1. Example `user_nl_clm` namelist file**

```
!-----------------------------------------------------------------------------
! Users should add all user specific namelist changes below in the form of
! namelist_var = new_namelist_value
!
! Include namelist variables for drv_flds_in ONLY if -megan and/or -drydep options
! are set in the CLM_NAMELIST_OPTS env variable.
!
! EXCEPTIONS:
```

```
! Set co2_ppmv           with CCSM_CO2_PPMV                     option
! Set dtime              with L_NCPL                            option
! Set fatmlndfrc         with LND_DOMAIN_PATH/LND_DOMAIN_FILE   options
! Set finidat            with RUN_REFCASE/RUN_REFDATE/RUN_REFTOD options for hybrid or branch cases
!                        (includes $inst_string for multi-ensemble cases)
! Set glc_grid           with GLC_GRID                          option
! Set glc_smb            with GLC_SMB                           option
! Set maxpatch_glcmec    with GLC_NEC                           option
!-----------------------------------------------------------------------------
 hist_fincl2    = 'TG','TBOT','FIRE','FIRA','FLDS','FSDS',
                  'FSR','FSA','FGEV','FSH','FGR','TSOI',
                  'ERRSOI','BUILDHEAT','SABV','SABG',
                  'FSDSVD','FSDSND','FSDSVI','FSDSNI',
                  'FSRVD','FSRND','FSRVI','FSRNI',
                  'TSA','FCTR','FCEV','QBOT','RH2M','H2OSOI',
                  'H2OSNO','SOILLIQ','SOILICE',
                  'TSA_U', 'TSA_R',
                  'TREFMNAV_U', 'TREFMNAV_R',
                  'TREFMXAV_U', 'TREFMXAV_R',
                  'TG_U', 'TG_R',
                  'RH2M_U', 'RH2M_R',
                  'QRUNOFF_U', 'QRUNOFF_R',
                  'SoilAlpha_U',
                  'Qanth', 'SWup', 'LWup', 'URBAN_AC', 'URBAN_HEAT'
  hist_fincl3 = 'TG:I', 'FSA:I', 'SWup:I', 'URBAN_AC:I', 'URBAN_HEAT:I',
                'TG_U:I', 'TG_R:I',
  hist_fincl4 = 'TG', 'FSA', 'SWup', 'URBAN_AC', 'URBAN_HEAT'
  hist_mfilt  = 1, 30,  28, 24
  hist_nhtfrq = 0, -24, -6, -1
```

**Note:** The comments at the top are some guidance given in the default `user_nl_clm` and just give some guidance on how to set variables and use the file.

**Note:** See the the Section called *Precedence of Options* section for the precedence of this option relative to the others.

**Note:** You do NOT need to specify the namelist group that the variables are in because the CLM **build-namelist** knows the namelist that specific variable names belong to, and it puts them there.

Obviously, all of this would be difficult to put in the CLM_NAMELIST_OPTS variable, especially having to put ' around all the character strings. For more information on the namelist variables being set here and what they mean, see the section on CLM namelists below, as well as the namelist definition that gives details on each variable.

## Precedence of Options

Note: The precedence for setting the values of namelist variables with the different `env_build.xml`, `env_run.xml` options is (highest to lowest):

1. Namelist values set by specific command-line options, like, -d, -sim_year (i.e. CLM_BLDNML_OPTS `env_build.xml` variable)
2. Values set on the command-line using the -namelist option, (i.e. CLM_NAMELIST_OPTS `env_run.xml` variable)
3. Values read from the file specified by -infile, (i.e. `user_nl_clm` file)
4. Datasets from the -clm_usr_name option, (i.e. CLM_USRDAT_NAME `env_run.xml` variable)
5. Values set from a use-case scenario, e.g., -use_case (i.e. CLM_NML_USE_CASE `env_run.xml` variable)
6. Values from the namelist defaults file.

Thus a setting in CLM_BLDNML_OPTS will override a setting for the same thing given in a use case with CLM_NML_USE_CASE. Likewise, a setting in CLM_NAMELIST_OPTS will override a setting in `user_nl_clm`.

## Setting Your Initial Conditions File

Especially with CLMCN starting from initial conditions is very important. Even with CLMSP it takes many simulation years to get the model fully spunup. There are a couple different ways to provide an initial condition file.

the Section called *Doing a hybrid simulation to provide initial conditions*
the Section called *Doing a branch simulation to provide initial conditions*
the Section called *Providing a finidat file in your `user_nl_clm` file*
the Section called *Adding a finidat file to the XML database*

> **Note:** Your initial condition file MUST agree with the surface dataset you are using to run the simulation. If the two files do NOT agree you will get a run-time about a mis-match in PFT weights, or in the number of PFT's or columns. To get around this you'll need to use the Section called *Using **interpinic** to interpolate initial conditions to different resolutions* in Chapter 2 to interpolate your initial condition dataset.

### Doing a hybrid simulation to provide initial conditions

The first option is to setup a hybrid simulation and give a RUN_REFCASE and RUN_REFDATE to specify the reference case simulation name to use. When you setup most cases, at the standard resolutions of "f09" or "f19" it will already do this for you. For example, if you run an "I2000CN"

compset at "f09_g16" resolution the following settings will already be done for you.

```
./xmlchange RUN_TYPE=hybrid,RUN_REFCASE=I2000CN_f09_g16_c100503,RUN_REFDATE=0001-01-01,GET_REFCASE=TRUE
```

Setting the GET_REFCASE option to `TRUE` means it will copy the files from the: `$DIN_LOC_ROOT/ccsm4_init/I2000CN_f09_g16_c100503/0001-01-01` directory. Note, that the `RUN_REFCASE` and `RUN_REFDATE` variables are expanded to get the directory name above. If you do NOT set GET_REFCASE to `TRUE` then you will need to have placed the file in your run directory yourself. In either case, the file is expected to be named: `$RUN_REFCASE.clm2.r.$RUN_REFDATE-00000.nc` with the variables expanded of course.

### Doing a branch simulation to provide initial conditions

The setup for running a branch simulation is essentially the same as for a hybrid. With the exception of setting RUN_TYPE to `branch` rather than `hybrid`. A branch simulation runs the case essentially as restarting from it's place before to exactly reproduce it (but possibly output more or different fields on the history files). While a hybrid simulation allows you to change the configuration or run-time options, as well as use a different code base than the original case that may have fewer fields on it than a full restart file. The GET_REFCASE option works similarily for a branch case as for a hybrid.

### Providing a finidat file in your `user_nl_clm` file

Setting up a branch or hybrid simulation requires the initial condition file to follow a standard naming convention, and a standard input directory if you use the GET_REFCASE option. If you want to name your file willy nilly and place it anywhere, you can set it in your `user_nl_clm` file as in this example.

```
finidat    = '/glade/home/$USER/myinitdata/clmi_I1850CN_f09_g16_0182-01-01.c120329.nc'
```

Note, if you provide an initial condition file -- you can NOT set CLM_FORCE_COLDSTART to `TRUE`.

### Adding a finidat file to the XML database

Like other datasets, if you want to use a given initial condition file to be used for all (or most of) your cases you'll want to put it in the XML database so it will be used by default. The initial condition files, are resolution dependent, and dependent on the number of PFT's and other variables such as GLC_NEC or if irrigation is on or off. See Chapter 3 for more information on this.

## Other noteworthy configuration items

For running "I" cases there are several other noteworthy configuration items that you may want to work with. Most of these involve settings for the DATM, but one CCSM_CO2_PPMV applies to all models. If

you are running an B, E, or F case that doesn't use the DATM obviously the DATM_* settings will not be used. All of the settings below are in your `env_build.xml` and `env_run.xml` files

CCSM_CO2_PPMV
CCSM_VOC
DATM_MODE
DATM_PRESAERO
DATM_CLMNCEP_YR_ALIGN
DATM_CLMNCEP_YR_START
DATM_CLMNCEP_YR_END
DATM_CPL_CASE
DATM_CPL_YR_ALIGN
DATM_CPL_YR_START
DATM_CPL_YR_END

CCSM_CO2_PPMV

CCSM_CO2_PPMV sets the mixing ratio of $CO_2$ in parts per million by volume for ALL CESM components to use. Note that most compsets already set this value to something reasonable. Also note that some compsets may tell the atmosphere model to override this value with either historic or ramped values. If the CCSM_BGC variable is set to something other than "none" the atmosphere model will determine $CO_2$, and CLM will listen and use what the atmosphere sends it. On the CLM side the namelist item `co2_type` tells CLM to use the value sent from the atmosphere rather than a value set on it's own namelist.

CCSM_VOC

CCSM_VOC enables passing of the Volatile Organic Compounds (VOC) from CLM to the atmospheric model. This of course is only important if the atmosphere model is a fully active model that can use these fields in it's chemistry calculations.

DATM_MODE

DATM_MODE sets the mode that the DATM model should run in this determines how data is handled as well as what the source of the data will be. Many of the modes are setup specifically to be used for ocean and/or sea-ice modeling. The modes that are designed for use by CLM are:

`CLM_QIAN`
`CLM1PT`>
`CPLHIST3HrWx`

`CLM_QIAN` is for the standard mode of using global atmospheric data that was developed by Qian et. al. for CLM using NCEP data from 1948 to 2004. See the Section called *`CLM_QIAN` mode and it's DATM settings* for more information on the DATM settings for `CLM_QIAN` mode. `CLM1PT` is for the special cases where we have single-point tower data for particular sites. Right now we only have data for three urban locations: MexicoCity Mexico, Vancouver Canada, and the urban-c alpha site. And we have data for the US-UMB AmeriFlux tower site for Univeristy of Michigan Biological Station. See the Section called *`CLM1PT` mode and it's DATM settings* for more information on the DATM settings for `CLM1PT` mode. `CPLHIST3HrWx` is for running with atmospheric forcing from a previous CESM simulation. See the Section called *`CPLHIST3HrWx` mode and it's DATM settings* for more information on the DATM settings for `CPLHIST3HrWx` mode.

DATM_PRESAERO

DATM_PRESAERO sets the prescribed aerosol mode for the data atmosphere model. The list of

valid options include:
`clim_1850` = constant year 1850 conditions
`clim_2000` = constant year 2000 conditions
`trans_1850-2000` = transient 1850 to year 2000 conditions
`rcp2.6` = transient conditions for the rcp=2.6 W/m2 future scenario
`rcp4.5` = transient conditions for the rcp=4.5 W/m2 future scenario
`rcp6.0` = transient conditions for the rcp=6.0 W/m2 future scenario
`rcp8.5` = transient conditions for the rcp=8.5 W/m2 future scenario
`pt1_pt1` = read in single-point or regional datasets

DATM_CLMNCEP_YR_START

DATM_CLMNCEP_YR_START sets the beginning year to cycle the atmospheric data over for the `CLM_QIAN` mode.

DATM_CLMNCEP_YR_END

DATM_CLMNCEP_YR_END sets the ending year to cycle the atmospheric data over for the `CLM_QIAN` mode.

DATM_CLMNCEP_YR_ALIGN

DATM_CLMNCEP_YR_START and DATM_CLMNCEP_YR_END determine the range of years to cycle the atmospheric data over, and DATM_CLMNCEP_YR_ALIGN determines which year in that range of years the simulation will start with.

DATM_CPL_CASE

DATM_CPL_CASE sets the casename to use for the `CPLHIST3HrWx` mode.

DATM_CPL_YR_START

DATM_CPL_YR_START sets the beginning year to cycle the atmospheric data over for the `CPLHIST3HrWx` mode.

DATM_CPL_YR_END

DATM_CPL_YR_END sets the ending year to cycle the atmospheric data over for the `CPLHIST3HrWx` mode.

DATM_CPL_YR_ALIGN

DATM_CPL_YR_START and DATM_CPL_YR_END determine the range of years to cycle the atmospheric data over, and DATM_CPL_YR_ALIGN determines which year in that range of years the simulation will start with.

## Downloading DATM Forcing Data

In Chapter One of the CESM User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm/cesm_doc/book1.html) there is a section on "Downloading input data". The normal process of setting up cases will use the "scripts/ccsm_utils/Tools/check_input_data" script to retrieve data from the CESM subversion inputdata repository. This is true for the standard `CLM_QIAN` forcing as well. However, the `CPLHIST3HrWx` DATM forcing data is unique -- because it is large compared to the rest of the input data, and we only have a

disk copy on yellowstone. The DATM assumes the path for the previous NCAR machine bluefire of `/glade/data01/CMIP5/CCSM/csm` for the data. So you will need to change this path in order to run on any machines. You can download the data itself from NCAR HPSS from `/CCSM/csm/$DATM_CPLHIST_CASE`.

## Customizing via the build script files

The final thing that the user may wish to do before **cesm_setup** is run is to edit the build script files which determine the configuration and namelist. The variables in `env_build.xml` or `env_run.xml` typically mean you will NOT have to edit build script files. But, there are rare instances where it is useful to do so. The build script files are copied to your case directory and are available under `Buildconf`. The list of build script files you might wish to edit are:

**clm.buildexe.csh**
**clm.buildnml.csh**
**datm.buildexe.csh**
**datm.buildnml.csh**

## More information on the CLM configure script

The CLM **configure** script defines the details of a clm configuration and summarizes it into a `config_cache.xml` file. The `config_cache.xml` will be placed in your case directory under `Buildconf/clmconf`. The config_definition.xml (../../bld/config_files/config_definition.xml) in `models/lnd/clm/bld/config_files` gives a definition of each CLM configuration item, it is viewable in a web-browser. Many of these items are things that you would NOT change, but looking through the list gives you the valid options, and a good description of each. Below we repeat the `config_definition.xml` files contents:

### Help on CLM configure

Coupling this with looking at the options to CLM **configure** with "-help" as below will enable you to understand how to set the different options.

```
> cd models/lnd/clm/bld
> configure -help
```

The output to the above command is as follows:

```
SYNOPSIS
     configure [options]

     Configure CLM in preparation to be built.
OPTIONS
     User supplied values are denoted in angle brackets (<>).  Any value that contains
     white-space must be quoted.  Long option names may be supplied with either single
     or double leading dashes.  A consequence of this is that single letter options may
     NOT be bundled.

     -bgc <name>            Build CLM with BGC package [ none | cn | cndv ]
```

```
                        (default is none).
  -c13 <name>           Turn on C13 mode for BGC setting of CN [on | off]
                        (default is off).
  -cache <file>         Name of output cache file (default: config_cache.xml).
  -cachedir <file>      Name of directory where output cache file is written
                        (default: CLM build directory).
  -clm_root <dir>       Root directory of clm source code
                        (default: directory above location of this script)
  -cppdefs <string>     A string of user specified CPP defines.  Appended to
                        Makefile defaults.  e.g. -cppdefs '-DVAR1 -DVAR2'
  -crop <name>          Toggle for prognostic crop model. [on | off] (default is off)
                        (can ONLY be turned on when BGC type is CN or CNDV)
  -comp_intf <name>     Component interface to use (ESMF or MCT) (default MCT)
  -defaults <file>      Specify full path to a configuration file which will be used
                        to supply defaults instead of the defaults in bld/config_files.
                        This file is used to specify model configuration parameters only.
                        Parameters relating to the build which are system dependent will
                        be ignored.
  -help [or -h]         Print usage to STDOUT.
  -nofire               Turn off wildfires for BGC setting of CN
                        (default includes fire for CN)
  -noio                 Turn history output completely off (typically for testing).
  -pergro <name>        Switch enables building CLM for perturbation growth tests. [on |  \
off]
                        (default is off)
  -silent [or -s]       Turns on silent mode - only fatal messages issued.
  -sitespf_pt <name>    Setup for the given site specific single-point resolution.
  -snicar_frc <name>    Turn on SNICAR radiative forcing calculation. [on | off]
                        (default is off)
  -spinup <name>        Turn on given spinup mode for BGC setting of CN    (level)
                          AD            Turn on Accelerated Decomposition from       (5)
                                        bare-soil
                          exit          Jump directly from AD spinup to normal mode       (2)
                          normal        Normal decomposition ("final spinup mode")       (0)
                                        (default)
                        Valid sequence: 5-2-0
  -usr_src <dir1>[,<dir2>[,<dir3>[...]]]
                        Directories containing user source code.
  -verbose [or -v]      Turn on verbose echoing of settings made by configure.
  -version              Echo the SVN tag name used to check out this CLM distribution.
```

We've given details on how to use the options in `env_build.xml` and `env_run.xml` to interact with the CLM "**configure**" and "**build-namelist**" scripts, as well as giving a good understanding of how these scripts work and the options to them. In the next section we give further details on the CLM namelist. You could customize the namelist for these options after "**cesm_setup**" is run.

# Customizing the CLM namelist

Once a case is **cesm_setup**, we can then customize the case further, by editing the run-time namelist for CLM. First let's list the definition of each namelist item and their valid values, and then we'll list the default values for them. Next for some of the most used or tricky namelist items we'll give examples of their use, and give you example namelists that highlight these features.

## Definition of Namelist items and their default values

Here we point to you where you can find the definition of each namelist item and separately the default values for them. The default values may change depending on the resolution, land-mask, simulation-year and other attributes. Both of these files are viewable in your web browser, and then expand each in turn.

1. Definition of Namelists Relevant for CLM (modelnl/index.html)
2. Default values of each CLM Namelist Item (../../bld/namelist_files/namelist_defaults_clm.xml)

One set of the namelist items allows you to add fields to the output history files: `hist_fincl1`, `hist_fincl2`, `hist_fincl3`, `hist_fincl4`, `hist_fincl5`, and `hist_fincl6`. The link CLM History Fields (../../bld/namelist_files/history_fields.xml) documents all of the history fields available and gives the long-name and units for each.

### Definition of CLM history variables

Included in the table are the following pieces of information:

- Variable name.
- Long name description.
- units

**Table 1-1. CLM History Fields**

| Name | Long-name | Units |
|---|---|---|
| **A10TMIN** | 10-day running mean of min 2-m temperature | K |
| **A5TMIN** | 5-day running mean of min 2-m temperature | K |
| **ACTUAL_IMMOB** | actual N immobilization | gN/m^2/s |
| **AGDD** | growing degree-days base 5C | K |
| **AGNPP** | aboveground NPP | gC/m^2/s |
| **ALBD** | surface albedo (direct) | proportion |
| **ALBGRD** | ground albedo (direct) | proportion |
| **ALBGRI** | ground albedo (indirect) | proportion |
| **ALBI** | surface albedo (indirect) | proportion |
| **ALLOC_PNOW** | fraction of current allocation to display as new growth | proportion |
| **ALPHA** | alpha coefficient for VOC calc | non |
| **ALPHAPSNSHA** | shaded c13 fractionation | proportion |
| **ALPHAPSNSUN** | sunlit c13 fractionation | proportion |
| **ANNAVG_T2M** | annual average 2m air temperature | K |

| Name | Long-name | Units |
|---|---|---|
| **ANNMAX_RETRANSN** | annual max of retranslocated N pool | gN/m^2 |
| **ANNSUM_COUNTER** | seconds since last annual accumulator turnover | s |
| **ANNSUM_NPP** | annual sum of NPP | gC/m^2/yr |
| **ANNSUM_POTENTIAL_GPP** | annual sum of potential GPP | gN/m^2/yr |
| **ANN_FAREA_BURNED** | annual total fractional area burned | proportion |
| **AR** | autotrophic respiration (MR + GR) | gC/m^2/s |
| **AVAILC** | C flux available for allocation | gC/m^2/s |
| **AVAIL_RETRANSN** | N flux available from retranslocation pool | gN/m^2/s |
| **BCDEP** | total BC deposition (dry+wet) from atmosphere | kg/m^2/s |
| **BETA** | coefficient of convective velocity | none |
| **BGLFR** | background litterfall rate | 1/s |
| **BGNPP** | belowground NPP | gC/m^2/s |
| **BGTR** | background transfer growth rate | 1/s |
| **BTRAN** | transpiration beta factor | unitless |
| **BUILDHEAT** | heat flux from urban building interior to walls and roof | W/m^2 |
| **C13_AGNPP** | C13 aboveground NPP | gC13/m^2/s |
| **C13_AR** | C13 autotrophic respiration (MR + GR) | gC13/m^2/s |
| **C13_BGNPP** | C13 belowground NPP | gC13/m^2/s |
| **C13_COL_CTRUNC** | C13 column-level sink for C truncation | gC13/m^2 |
| **C13_COL_FIRE_CLOSS** | C13 total column-level fire C loss | gC13/m^2/s |
| **C13_CPOOL** | C13 temporary photosynthate C pool | gC13/m^2 |
| **C13_CPOOL_DEADCROOT_GR** | C13 dead coarse root growth respiration | gC13/m^2/s |
| **C13_CPOOL_DEADCROOT_STORAGE_GR** | C13 dead coarse root growth respiration to storage | gC13/m^2/s |
| **C13_CPOOL_DEADSTEM_GR** | C13 dead stem growth respiration | gC13/m^2/s |
| **C13_CPOOL_DEADSTEM_STORAGE_GR** | C13 dead stem growth respiration to storage | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_CPOOL_FROOT_GR** | C13 fine root growth respiration | gC13/m^2/s |
| **C13_CPOOL_FROOT_STORAGE_GR** | C13 fine root growth respiration to storage | gC13/m^2/s |
| **C13_CPOOL_LEAF_GR** | C13 leaf growth respiration | gC13/m^2/s |
| **C13_CPOOL_LEAF_STORAGE_GR** | C13 leaf growth respiration to storage | gC13/m^2/s |
| **C13_CPOOL_LIVECROOT_GR** | C13 live coarse root growth respiration | gC13/m^2/s |
| **C13_CPOOL_LIVECROOT_STORAGE_GR** | C13 live coarse root growth respiration to storage | gC13/m^2/s |
| **C13_CPOOL_LIVESTEM_GR** | C13 live stem growth respiration | gC13/m^2/s |
| **C13_CPOOL_LIVESTEM_STORAGE_GR** | C13 live stem growth respiration to storage | gC13/m^2/s |
| **C13_CPOOL_TO_DEADCROOTC** | C13 allocation to dead coarse root C | gC13/m^2/s |
| **C13_CPOOL_TO_DEADCROOTC_STORAGE** | C13 allocation to dead coarse root C storage | gC13/m^2/s |
| **C13_CPOOL_TO_DEADSTEMC** | C13 allocation to dead stem C | gC13/m^2/s |
| **C13_CPOOL_TO_DEADSTEMC_STORAGE** | C13 allocation to dead stem C storage | gC13/m^2/s |
| **C13_CPOOL_TO_FROOTC** | C13 allocation to fine root C | gC13/m^2/s |
| **C13_CPOOL_TO_FROOTC_STORAGE** | C13 allocation to fine root C storage | gC13/m^2/s |
| **C13_CPOOL_TO_GRESP_STORAGE** | C13 allocation to growth respiration storage | gC13/m^2/s |
| **C13_CPOOL_TO_LEAFC** | C13 allocation to leaf C | gC13/m^2/s |
| **C13_CPOOL_TO_LEAFC_STORAGE** | C13 allocation to leaf C storage | gC13/m^2/s |
| **C13_CPOOL_TO_LIVECROOTC** | C13 allocation to live coarse root C | gC13/m^2/s |
| **C13_CPOOL_TO_LIVECROOTC_STORAGE** | C13 allocation to live coarse root C storage | gC13/m^2/s |
| **C13_CPOOL_TO_LIVESTEMC** | C13 allocation to live stem C | gC13/m^2/s |
| **C13_CPOOL_TO_LIVESTEMC_STORAGE** | C13 allocation to live stem C storage | gC13/m^2/s |
| **C13_CURRENT_GR** | C13 growth resp for new growth displayed in this timestep | gC13/m^2/s |
| **C13_CWDC** | C13 coarse woody debris C | gC13/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **C13_CWDC_TO_LITR2C** | C13 decomp. of coarse woody debris C to litter 2 C | gC13/m^2/s |
| **C13_CWDC_TO_LITR3C** | C13 decomp. of coarse woody debris C to litter 3 C | gC13/m^2/s |
| **C13_DEADCROOTC** | C13 dead coarse root C | gC13/m^2 |
| **C13_DEADCROOTC_STORAGE** | C13 dead coarse root C storage | gC13/m^2 |
| **C13_DEADCROOTC_STORAGE_TO_XFER** | C13 dead coarse root C shift storage to transfer | gC13/m^2/s |
| **C13_DEADCROOTC_XFER** | C13 dead coarse root C transfer | gC13/m^2 |
| **C13_DEADCROOTC_XFER_TO_DEADCROOTC** | C13 dead coarse root C growth from storage | gC13/m^2/s |
| **C13_DEADSTEMC** | C13 dead stem C | gC13/m^2 |
| **C13_DEADSTEMC_STORAGE** | C13 dead stem C storage | gC13/m^2 |
| **C13_DEADSTEMC_STORAGE_TO_XFER** | C13 dead stem C shift storage to transfer | gC13/m^2/s |
| **C13_DEADSTEMC_XFER** | C13 dead stem C transfer | gC13/m^2 |
| **C13_DEADSTEMC_XFER_TO_DEADSTEMC** | C13 dead stem C growth from storage | gC13/m^2/s |
| **C13_DISPVEGC** | C13 displayed veg carbon, excluding storage and cpool | gC13/m^2 |
| **C13_DWT_CLOSS** | C13 total carbon loss from land cover conversion | gC13/m^2/s |
| **C13_DWT_CONV_CFLUX** | C13 conversion C flux (immediate loss to atm) | gC13/m^2/s |
| **C13_DWT_DEADCROOTC_TO_CWDC** | C13 dead coarse root to CWD due to landcover change | gC13/m^2/s |
| **C13_DWT_FROOTC_TO_LITR1C** | C13 fine root to litter due to landcover change | gC13/m^2/s |
| **C13_DWT_FROOTC_TO_LITR2C** | C13 fine root to litter due to landcover change | gC13/m^2/s |
| **C13_DWT_FROOTC_TO_LITR3C** | C13 fine root to litter due to landcover change | gC13/m^2/s |
| **C13_DWT_LIVECROOTC_TO_CWDC** | C13 live coarse root to CWD due to landcover change | gC13/m^2/s |
| **C13_DWT_PROD100C_GAIN** | C13 addition to 100-yr wood product pool | gC13/m^2/s |
| **C13_DWT_PROD10C_GAIN** | C13 addition to 10-yr wood product pool | gC13/m^2/s |
| **C13_DWT_SEEDC_TO_DEADSTEM** | C13 seed source to PFT-level deadstem | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_DWT_SEEDC_TO_LEAF** | C13 seed source to PFT-level leaf | gC13/m^2/s |
| **C13_ER** | C13 total ecosystem respiration, autotrophic + heterotrophic | gC13/m^2/s |
| **C13_FROOTC** | C13 fine root C | gC13/m^2 |
| **C13_FROOTC_STORAGE** | C13 fine root C storage | gC13/m^2 |
| **C13_FROOTC_STORAGE_TO_XFER** | C13 fine root C shift storage to transfer | gC13/m^2/s |
| **C13_FROOTC_TO_LITR1C** | C13 fine root C litterfall to litter 1 C | gC13/m^2/s |
| **C13_FROOTC_TO_LITR2C** | C13 fine root C litterfall to litter 2 C | gC13/m^2/s |
| **C13_FROOTC_TO_LITR3C** | C13 fine root C litterfall to litter 3 C | gC13/m^2/s |
| **C13_FROOTC_TO_LITTER** | C13 fine root C litterfall | gC13/m^2/s |
| **C13_FROOTC_XFER** | C13 fine root C transfer | gC13/m^2 |
| **C13_FROOTC_XFER_TO_FROOTC** | C13 fine root C growth from storage | gC13/m^2/s |
| **C13_FROOT_MR** | C13 fine root maintenance respiration | gC13/m^2/s |
| **C13_GPP** | C13 gross primary production | gC13/m^2/s |
| **C13_GR** | C13 total growth respiration | gC13/m^2/s |
| **C13_GRESP_STORAGE** | C13 growth respiration storage | gC13/m^2 |
| **C13_GRESP_STORAGE_TO_XFER** | C13 growth respiration shift storage to transfer | gC13/m^2/s |
| **C13_GRESP_XFER** | C13 growth respiration transfer | gC13/m^2 |
| **C13_HR** | C13 total heterotrophic respiration | gC13/m^2/s |
| **C13_LEAFC** | C13 leaf C | gC13/m^2 |
| **C13_LEAFC_STORAGE** | C13 leaf C storage | gC13/m^2 |
| **C13_LEAFC_STORAGE_TO_XFER** | C13 leaf C shift storage to transfer | gC13/m^2/s |
| **C13_LEAFC_TO_LITR1C** | C13 leaf C litterfall to litter 1 C | gC13/m^2/s |
| **C13_LEAFC_TO_LITR2C** | C13 leaf C litterfall to litter 2 C | gC13/m^2/s |
| **C13_LEAFC_TO_LITR3C** | C13 leaf C litterfall to litter 3 C | gC13/m^2/s |
| **C13_LEAFC_TO_LITTER** | C13 leaf C litterfall | gC13/m^2/s |
| **C13_LEAFC_XFER** | C13 leaf C transfer | gC13/m^2 |
| **C13_LEAFC_XFER_TO_LEAFC** | C13 leaf C growth from storage | gC13/m^2/s |
| **C13_LEAF_MR** | C13 leaf maintenance respiration | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_LITFALL** | C13 litterfall (leaves and fine roots) | gC13/m^2/s |
| **C13_LITFIRE** | C13 litter fire losses | gC13/m^2/s |
| **C13_LITHR** | C13 litter heterotrophic respiration | gC13/m^2/s |
| **C13_LITR1C** | C13 litter labile C | gC13/m^2 |
| **C13_LITR1C_TO_SOIL1C** | C13 decomp. of litter 1 C to SOM 1 C | gC13/m^2/s |
| **C13_LITR1_HR** | C13 het. resp. from litter 1 C | gC13/m^2/s |
| **C13_LITR2C** | C13 litter cellulose C | gC13/m^2 |
| **C13_LITR2C_TO_SOIL2C** | C13 decomp. of litter 2 C to SOM 2 C | gC13/m^2/s |
| **C13_LITR2_HR** | C13 het. resp. from litter 2 C | gC13/m^2/s |
| **C13_LITR3C** | C13 litter lignin C | gC13/m^2 |
| **C13_LITR3C_TO_SOIL3C** | C13 decomp. of litter 3 C to SOM 3 C | gC13/m^2/s |
| **C13_LITR3_HR** | C13 het. resp. from litter 3 C | gC13/m^2/s |
| **C13_LIVECROOTC** | C13 live coarse root C | gC13/m^2 |
| **C13_LIVECROOTC_STORAGE** | C13 live coarse root C storage | gC13/m^2 |
| **C13_LIVECROOTC_STORAGE_TO_XFER** | C13 live coarse root C shift storage to transfer | gC13/m^2/s |
| **C13_LIVECROOTC_TO_DEADCROOTC** | C13 live coarse root C turnover | gC13/m^2/s |
| **C13_LIVECROOTC_XFER** | C13 live coarse root C transfer | gC13/m^2 |
| **C13_LIVECROOTC_XFER_TO_LIVECROOTC** | C13 live coarse root C growth from storage | gC13/m^2/s |
| **C13_LIVECROOT_MR** | C13 live coarse root maintenance respiration | gC13/m^2/s |
| **C13_LIVESTEMC** | C13 live stem C | gC13/m^2 |
| **C13_LIVESTEMC_STORAGE** | C13 live stem C storage | gC13/m^2 |
| **C13_LIVESTEMC_STORAGE_TO_XFER** | C13 live stem C shift storage to transfer | gC13/m^2/s |
| **C13_LIVESTEMC_TO_DEADSTEMC** | C13 live stem C turnover | gC13/m^2/s |
| **C13_LIVESTEMC_XFER** | C13 live stem C transfer | gC13/m^2 |
| **C13_LIVESTEMC_XFER_TO_LIVESTEMC** | C13 live stem C growth from storage | gC13/m^2/s |
| **C13_LIVESTEM_MR** | C13 live stem maintenance respiration | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_MR** | C13 maintenance respiration | gC13/m^2/s |
| **C13_M_CWDC_TO_FIRE** | C13 coarse woody debris C fire loss | gC13/m^2/s |
| **C13_M_DEADCROOTC_STORAGE_TO_FIRE** | C13 dead coarse root C storage fire loss | gC13/m^2/s |
| **C13_M_DEADCROOTC_STORAGE_TO_LITR1C** | C13 dead coarse root C storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_DEADCROOTC_STORAGE_TO_LITTER** | C13 dead coarse root C storage mortality | gC13/m^2/s |
| **C13_M_DEADCROOTC_TO_CWDC** | C13 dead coarse root C mortality to coarse woody debris C | gC13/m^2/s |
| **C13_M_DEADCROOTC_TO_CWDC_FIRE** | C13 dead coarse root C to to woody debris C by fire | gC13/m^2/s |
| **C13_M_DEADCROOTC_TO_FIRE** | C13 dead coarse root C fire loss | gC13/m^2/s |
| **C13_M_DEADCROOTC_TO_LITTER** | C13 dead coarse root C mortality | gC13/m^2/s |
| **C13_M_DEADCROOTC_TO_LITTER_FIRE** | C13 dead coarse root C fire mortality to litter | gC13/m^2/s |
| **C13_M_DEADCROOTC_XFER_TO_FIRE** | C13 dead coarse root C transfer fire loss | gC13/m^2/s |
| **C13_M_DEADCROOTC_XFER_TO_LITR1C** | C13 dead coarse root C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_DEADCROOTC_XFER_TO_LITTER** | C13 dead coarse root C transfer mortality | gC13/m^2/s |
| **C13_M_DEADSTEMC_STORAGE_TO_FIRE** | C13 dead stem C storage fire loss | gC13/m^2/s |
| **C13_M_DEADSTEMC_STORAGE_TO_LITR1C** | C13 dead stem C storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_DEADSTEMC_STORAGE_TO_LITTER** | C13 dead stem C storage mortality | gC13/m^2/s |
| **C13_M_DEADSTEMC_TO_CWDC** | C13 dead stem C mortality to coarse woody debris C | gC13/m^2/s |
| **C13_M_DEADSTEMC_TO_CWDC_FIRE** | C13 dead stem C to coarse woody debris C by fire | gC13/m^2/s |
| **C13_M_DEADSTEMC_TO_FIRE** | C13 dead stem C fire loss | gC13/m^2/s |
| **C13_M_DEADSTEMC_TO_LITTER** | C13 dead stem C mortality | gC13/m^2/s |
| **C13_M_DEADSTEMC_TO_LITTER_FIRE** | C13 dead stem C fire mortality to litter | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_M_DEADSTEMC_XFER_TO_FIRE** | C13 dead stem C transfer fire loss | gC13/m^2/s |
| **C13_M_DEADSTEMC_XFER_TO_LITR1C** | C13 dead stem C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_DEADSTEMC_XFER_TO_LITTER** | C13 dead stem C transfer mortality | gC13/m^2/s |
| **C13_M_FROOTC_STORAGE_TO_FIRE** | C13 fine root C storage fire loss | gC13/m^2/s |
| **C13_M_FROOTC_STORAGE_TO_LITR1C** | C13 fine root C storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_FROOTC_STORAGE_TO_LITTER** | C13 fine root C storage mortality | gC13/m^2/s |
| **C13_M_FROOTC_TO_FIRE** | C13 fine root C fire loss | gC13/m^2/s |
| **C13_M_FROOTC_TO_LITR1C** | C13 fine root C mortality to litter 1 C | gC13/m^2/s |
| **C13_M_FROOTC_TO_LITR2C** | C13 fine root C mortality to litter 2 C | gC13/m^2/s |
| **C13_M_FROOTC_TO_LITR3C** | C13 fine root C mortality to litter 3 C | gC13/m^2/s |
| **C13_M_FROOTC_TO_LITTER** | C13 fine root C mortality | gC13/m^2/s |
| **C13_M_FROOTC_XFER_TO_FIRE** | C13 fine root C transfer fire loss | gC13/m^2/s |
| **C13_M_FROOTC_XFER_TO_LITR1C** | C13 fine root C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_FROOTC_XFER_TO_LITTER** | C13 fine root C transfer mortality | gC13/m^2/s |
| **C13_M_GRESP_STORAGE_TO_FIRE** | C13 growth respiration storage fire loss | gC13/m^2/s |
| **C13_M_GRESP_STORAGE_TO_LITR1C** | C13 growth respiration storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_GRESP_STORAGE_TO_LITTER** | C13 growth respiration storage mortality | gC13/m^2/s |
| **C13_M_GRESP_XFER_TO_FIRE** | C13 growth respiration transfer fire loss | gC13/m^2/s |
| **C13_M_GRESP_XFER_TO_LITR1C** | C13 growth respiration transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_GRESP_XFER_TO_LITTER** | C13 growth respiration transfer mortality | gC13/m^2/s |
| **C13_M_LEAFC_STORAGE_TO_FIRE** | C13 leaf C storage fire loss | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_M_LEAFC_STORAGE_TO_LITR1C** | C13 leaf C storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LEAFC_STORAGE_TO_LITTER** | C13 leaf C storage mortality | gC13/m^2/s |
| **C13_M_LEAFC_TO_FIRE** | C13 leaf C fire loss | gC13/m^2/s |
| **C13_M_LEAFC_TO_LITR1C** | C13 leaf C mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LEAFC_TO_LITR2C** | C13 leaf C mortality to litter 2 C | gC13/m^2/s |
| **C13_M_LEAFC_TO_LITR3C** | C13 leaf C mortality to litter 3 C | gC13/m^2/s |
| **C13_M_LEAFC_TO_LITTER** | C13 leaf C mortality | gC13/m^2/s |
| **C13_M_LEAFC_XFER_TO_FIRE** | C13 leaf C transfer fire loss | gC13/m^2/s |
| **C13_M_LEAFC_XFER_TO_LITR1C** | C13 leaf C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LEAFC_XFER_TO_LITTER** | C13 leaf C transfer mortality | gC13/m^2/s |
| **C13_M_LITR1C_TO_FIRE** | C13 litter 1 C fire loss | gC13/m^2/s |
| **C13_M_LITR2C_TO_FIRE** | C13 litter 2 C fire loss | gC13/m^2/s |
| **C13_M_LITR3C_TO_FIRE** | C13 litter 3 C fire loss | gC13/m^2/s |
| **C13_M_LIVECROOTC_STORAGE_TO_FIRE** | C13 live coarse root C storage fire loss | gC13/m^2/s |
| **C13_M_LIVECROOTC_STORAGE_TO_LITR1C** | C13 live coarse root C storage mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LIVECROOTC_STORAGE_TO_LITTER** | C13 live coarse root C storage mortality | gC13/m^2/s |
| **C13_M_LIVECROOTC_TO_CWDC** | C13 live coarse root C mortality to coarse woody debris C | gC13/m^2/s |
| **C13_M_LIVECROOTC_TO_FIRE** | C13 live coarse root C fire loss | gC13/m^2/s |
| **C13_M_LIVECROOTC_TO_LITTER** | C13 live coarse root C mortality | gC13/m^2/s |
| **C13_M_LIVECROOTC_XFER_TO_FIRE** | C13 live coarse root C transfer fire loss | gC13/m^2/s |
| **C13_M_LIVECROOTC_XFER_TO_LITR1C** | C13 live coarse root C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LIVECROOTC_XFER_TO_LITTER** | C13 live coarse root C transfer mortality | gC13/m^2/s |
| **C13_M_LIVESTEMC_STORAGE_TO_FIRE** | C13 live stem C storage fire loss | gC13/m^2/s |
| **C13_M_LIVESTEMC_STORAGE_TO_LITTER** | C13 live stem C storage mortality to litter 1 C | gC13/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **C13_M_LIVESTEMC_STORAGE_TO_LITTER** | C13 live stem C storage mortality | gC13/m^2/s |
| **C13_M_LIVESTEMC_TO_CWDC** | C13 live stem C mortality to coarse woody debris C | gC13/m^2/s |
| **C13_M_LIVESTEMC_TO_FIRE** | C13 live stem C fire loss | gC13/m^2/s |
| **C13_M_LIVESTEMC_TO_LITTER** | C13 live stem C mortality | gC13/m^2/s |
| **C13_M_LIVESTEMC_XFER_TO_FIRE** | C13 live stem C transfer fire loss | gC13/m^2/s |
| **C13_M_LIVESTEMC_XFER_TO_LITR1C** | C13 live stem C transfer mortality to litter 1 C | gC13/m^2/s |
| **C13_M_LIVESTEMC_XFER_TO_LITTER** | C13 live stem C transfer mortality | gC13/m^2/s |
| **C13_NEE** | C13 net ecosystem exchange of carbon, includes fire flux, positive for source | gC13/m^2/s |
| **C13_NEP** | C13 net ecosystem production, excludes fire flux, positive for sink | gC13/m^2/s |
| **C13_NPP** | C13 net primary production | gC13/m^2/s |
| **C13_PFT_CTRUNC** | C13 pft-level sink for C truncation | gC13/m^2 |
| **C13_PFT_FIRE_CLOSS** | C13 total pft-level fire C loss | gC13/m^2/s |
| **C13_PROD100C** | C13 100-yr wood product C | gC13/m^2 |
| **C13_PROD100C_LOSS** | C13 loss from 100-yr wood product pool | gC13/m^2/s |
| **C13_PROD10C** | C13 10-yr wood product C | gC13/m^2 |
| **C13_PROD10C_LOSS** | C13 loss from 10-yr wood product pool | gC13/m^2/s |
| **C13_PRODUCT_CLOSS** | C13 total carbon loss from wood product pools | gC13/m^2/s |
| **C13_PSNSHA** | C13 shaded leaf photosynthesis | umolCO2/m^2/s |
| **C13_PSNSHADE_TO_CPOOL** | C13 C fixation from shaded canopy | gC13/m^2/s |
| **C13_PSNSUN** | C13 sunlit leaf photosynthesis | umolCO2/m^2/s |
| **C13_PSNSUN_TO_CPOOL** | C13 C fixation from sunlit canopy | gC13/m^2/s |
| **C13_RR** | C13 root respiration (fine root MR + total root GR) | gC13/m^2/s |
| **C13_SEEDC** | C13 pool for seeding new PFTs | gC13/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **C13_SOIL1C** | C13 soil organic matter C (fast pool) | gC13/m^2 |
| **C13_SOIL1C_TO_SOIL2C** | C13 decomp. of SOM 1 C to SOM 2 C | gC13/m^2/s |
| **C13_SOIL1_HR** | C13 het. resp. from SOM 1 C | gC13/m^2/s |
| **C13_SOIL2C** | C13 soil organic matter C (medium pool) | gC13/m^2 |
| **C13_SOIL2C_TO_SOIL3C** | C13 decomp. of SOM 2 C to SOM 3 C | gC13/m^2/s |
| **C13_SOIL2_HR** | C13 het. resp. from SOM 2 C | gC13/m^2/s |
| **C13_SOIL3C** | C13 soil organic matter C (slow pool) | gC13/m^2 |
| **C13_SOIL3C_TO_SOIL4C** | C13 decomp. of SOM 3 C to SOM 4 C | gC13/m^2/s |
| **C13_SOIL3_HR** | C13 het. resp. from SOM 3 C | gC13/m^2/s |
| **C13_SOIL4C** | C13 soil organic matter C (slowest pool) | gC13/m^2 |
| **C13_SOIL4_HR** | C13 het. resp. from SOM 4 C | gC13/m^2/s |
| **C13_SOMFIRE** | C13 soil organic matter fire losses | gC13/m^2/s |
| **C13_SOMHR** | C13 soil organic matter heterotrophic respiration | gC13/m^2/s |
| **C13_SR** | C13 total soil respiration (HR + root resp) | gC13/m^2/s |
| **C13_STORAGE_GR** | C13 growth resp for growth sent to storage for later display | gC13/m^2/s |
| **C13_STORVEGC** | C13 stored vegetation carbon, excluding cpool | gC13/m^2 |
| **C13_TOTCOLC** | C13 total column carbon, incl veg and cpool | gC13/m^2 |
| **C13_TOTECOSYSC** | C13 total ecosystem carbon, incl veg but excl cpool | gC13/m^2 |
| **C13_TOTFIRE** | C13 total ecosystem fire losses | gC13/m^2/s |
| **C13_TOTLITC** | C13 total litter carbon | gC13/m^2 |
| **C13_TOTPFTC** | C13 total pft-level carbon, including cpool | gC13/m^2 |
| **C13_TOTPRODC** | C13 total wood product C | gC13/m^2 |
| **C13_TOTSOMC** | C13 total soil organic matter carbon | gC13/m^2 |
| **C13_TOTVEGC** | C13 total vegetation carbon, excluding cpool | gC13/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **C13_TRANSFER_DEADCROOT_GR** | C13 dead coarse root growth respiration from storage | gC13/m^2/s |
| **C13_TRANSFER_DEADSTEM_GR** | C13 dead stem growth respiration from storage | gC13/m^2/s |
| **C13_TRANSFER_FROOT_GR** | C13 fine root growth respiration from storage | gC13/m^2/s |
| **C13_TRANSFER_GR** | C13 growth resp for transfer growth displayed in this timestep | gC13/m^2/s |
| **C13_TRANSFER_LEAF_GR** | C13 leaf growth respiration from storage | gC13/m^2/s |
| **C13_TRANSFER_LIVECROOT_GR** | C13 live coarse root growth respiration from storage | gC13/m^2/s |
| **C13_TRANSFER_LIVESTEM_GR** | C13 live stem growth respiration from storage | gC13/m^2/s |
| **C13_VEGFIRE** | C13 pft-level fire loss | gC13/m^2/s |
| **C13_XSMRPOOL** | C13 temporary photosynthate C pool | gC13/m^2 |
| **CANNAVG_T2M** | annual average of 2m air temperature | K |
| **CANNSUM_NPP** | annual sum of column-level NPP | gC/m^2/s |
| **CGRND** | deriv. of soil energy flux wrt to soil temp | W/m^2/K |
| **CGRNDL** | deriv. of soil latent heat flux wrt soil temp | W/m^2/K |
| **CGRNDS** | deriv. of soil sensible heat flux wrt soil temp | W/m^2/K |
| **CISHA** | shaded intracellular CO2 | Pa |
| **CISUN** | sunlit intracellular CO2 | Pa |
| **COL_CTRUNC** | column-level sink for C truncation | gC/m^2 |
| **COL_FIRE_CLOSS** | total column-level fire C loss | gC/m^2/s |
| **COL_FIRE_NLOSS** | total column-level fire N loss | gN/m^2/s |
| **COL_NTRUNC** | column-level sink for N truncation | gN/m^2 |
| **COSZEN** | cosine of solar zenith angle | none |
| **CP** | cp coefficient for VOC calc | non |
| **CPOOL** | temporary photosynthate C pool | gC/m^2 |
| **CPOOL_DEADCROOT_GR** | dead coarse root growth respiration | gC/m^2/s |
| **CPOOL_DEADCROOT_STORAGE_GR** | dead coarse root growth respiration to storage | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **CPOOL_DEADSTEM_GR** | dead stem growth respiration | gC/m^2/s |
| **CPOOL_DEADSTEM_STORAGE_GR** | dead stem growth respiration to storage | gC/m^2/s |
| **CPOOL_FROOT_GR** | fine root growth respiration | gC/m^2/s |
| **CPOOL_FROOT_STORAGE_GR** | fine root growth respiration to storage | gC/m^2/s |
| **CPOOL_LEAF_GR** | leaf growth respiration | gC/m^2/s |
| **CPOOL_LEAF_STORAGE_GR** | leaf growth respiration to storage | gC/m^2/s |
| **CPOOL_LIVECROOT_GR** | live coarse root growth respiration | gC/m^2/s |
| **CPOOL_LIVECROOT_STORAGE_GR** | live coarse root growth respiration to storage | gC/m^2/s |
| **CPOOL_LIVESTEM_GR** | live stem growth respiration | gC/m^2/s |
| **CPOOL_LIVESTEM_STORAGE_GR** | live stem growth respiration to storage | gC/m^2/s |
| **CPOOL_TO_DEADCROOTC** | allocation to dead coarse root C | gC/m^2/s |
| **CPOOL_TO_DEADCROOTC_STORAGE** | allocation to dead coarse root C storage | gC/m^2/s |
| **CPOOL_TO_DEADSTEMC** | allocation to dead stem C | gC/m^2/s |
| **CPOOL_TO_DEADSTEMC_STORAGE** | allocation to dead stem C storage | gC/m^2/s |
| **CPOOL_TO_FROOTC** | allocation to fine root C | gC/m^2/s |
| **CPOOL_TO_FROOTC_STORAGE** | allocation to fine root C storage | gC/m^2/s |
| **CPOOL_TO_GRESP_STORAGE** | allocation to growth respiration storage | gC/m^2/s |
| **CPOOL_TO_LEAFC** | allocation to leaf C | gC/m^2/s |
| **CPOOL_TO_LEAFC_STORAGE** | allocation to leaf C storage | gC/m^2/s |
| **CPOOL_TO_LIVECROOTC** | allocation to live coarse root C | gC/m^2/s |
| **CPOOL_TO_LIVECROOTC_STORAGE** | allocation to live coarse root C storage | gC/m^2/s |
| **CPOOL_TO_LIVESTEMC** | allocation to live stem C | gC/m^2/s |
| **CPOOL_TO_LIVESTEMC_STORAGE** | allocation to live stem C storage | gC/m^2/s |
| **CURRENT_GR** | growth resp for new growth displayed in this timestep | gC/m^2/s |
| **CWDC** | coarse woody debris C | gC/m^2 |
| **CWDC_HR** | coarse woody debris C heterotrophic respiration | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **CWDC_LOSS** | coarse woody debris C loss | gC/m^2/s |
| **CWDC_TO_LITR2C** | decomp. of coarse woody debris C to litter 2 C | gC/m^2/s |
| **CWDC_TO_LITR3C** | decomp. of coarse woody debris C to litter 3 C | gC/m^2/s |
| **CWDN** | coarse woody debris N | gN/m^2 |
| **CWDN_TO_LITR2N** | decomp. of coarse woody debris N to litter 2 N | gN/m^2/s |
| **CWDN_TO_LITR3N** | decomp. of coarse woody debris N to litter 3 N | gN/m^2/s |
| **C_ALLOMETRY** | C allocation index | none |
| **DAYL** | daylength | s |
| **DAYS_ACTIVE** | number of days since last dormancy | days |
| **DEADCROOTC** | dead coarse root C | gC/m^2 |
| **DEADCROOTC_STORAGE** | dead coarse root C storage | gC/m^2 |
| **DEADCROOTC_STORAGE_TO_XFER** | dead coarse root C shift storage to transfer | gC/m^2/s |
| **DEADCROOTC_XFER** | dead coarse root C transfer | gC/m^2 |
| **DEADCROOTC_XFER_TO_DEADCROOTC** | dead coarse root C growth from storage | gC/m^2/s |
| **DEADCROOTN** | dead coarse root N | gN/m^2 |
| **DEADCROOTN_STORAGE** | dead coarse root N storage | gN/m^2 |
| **DEADCROOTN_STORAGE_TO_XFER** | dead coarse root N shift storage to transfer | gN/m^2/s |
| **DEADCROOTN_XFER** | dead coarse root N transfer | gN/m^2 |
| **DEADCROOTN_XFER_TO_DEADCROOTN** | dead coarse root N growth from storage | gN/m^2/s |
| **DEADSTEMC** | dead stem C | gC/m^2 |
| **DEADSTEMC_STORAGE** | dead stem C storage | gC/m^2 |
| **DEADSTEMC_STORAGE_TO_XFER** | dead stem C shift storage to transfer | gC/m^2/s |
| **DEADSTEMC_XFER** | dead stem C transfer | gC/m^2 |
| **DEADSTEMC_XFER_TO_DEADSTEMC** | dead stem C growth from storage | gC/m^2/s |
| **DEADSTEMN** | dead stem N | gN/m^2 |
| **DEADSTEMN_STORAGE** | dead stem N storage | gN/m^2 |
| **DEADSTEMN_STORAGE_TO_XFER** | dead stem N shift storage to transfer | gN/m^2/s |
| **DEADSTEMN_XFER** | dead stem N transfer | gN/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **DEADSTEMN_XFER_TO_DEADSTEMN** | dead stem N growth from storage | gN/m^2/s |
| **DECL** | solar declination angle | radians |
| **DENIT** | total rate of denitrification | gN/m^2/s |
| **DEWMX** | Maximum allowed dew | mm |
| **DGNETDT** | derivative of net ground heat flux wrt soil temp | W/m^2/K |
| **DISPLA** | displacement height | m |
| **DISPVEGC** | displayed veg carbon, excluding storage and cpool | gC/m^2 |
| **DISPVEGN** | displayed vegetation nitrogen | gN/m^2 |
| **DLRAD** | downward longwave radiation below the canopy | W/m^2 |
| **DORMANT_FLAG** | dormancy flag | none |
| **DOWNREG** | fractional reduction in GPP due to N limitation | proportion |
| **DPVLTRB1** | turbulent deposition velocity 1 | m/s |
| **DPVLTRB2** | turbulent deposition velocity 2 | m/s |
| **DPVLTRB3** | turbulent deposition velocity 3 | m/s |
| **DPVLTRB4** | turbulent deposition velocity 4 | m/s |
| **DSTDEP** | total dust deposition (dry+wet) from atmosphere | kg/m^2/s |
| **DSTFLXT** | total surface dust emission | kg/m2/s |
| **DT_VEG** | change in t_veg, last iteration | K |
| **DWT_CLOSS** | total carbon loss from land cover conversion | gC/m^2/s |
| **DWT_CONV_CFLUX** | conversion C flux (immediate loss to atm) | gC/m^2/s |
| **DWT_CONV_NFLUX** | conversion N flux (immediate loss to atm) | gN/m^2/s |
| **DWT_DEADCROOTC_TO_CWDC** | dead coarse root to CWD due to landcover change | gC/m^2/s |
| **DWT_DEADCROOTN_TO_CWDN** | dead coarse root to CWD due to landcover change | gN/m^2/s |
| **DWT_FROOTC_TO_LITR1C** | fine root to litter due to landcover change | gC/m^2/s |
| **DWT_FROOTC_TO_LITR2C** | fine root to litter due to landcover change | gC/m^2/s |
| **DWT_FROOTC_TO_LITR3C** | fine root to litter due to landcover change | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **DWT_FROOTN_TO_LITR1N** | fine root to litter due to landcover change | gN/m^2/s |
| **DWT_FROOTN_TO_LITR2N** | fine root to litter due to landcover change | gN/m^2/s |
| **DWT_FROOTN_TO_LITR3N** | fine root to litter due to landcover change | gN/m^2/s |
| **DWT_LIVECROOTC_TO_CWDC** | live coarse root to CWD due to landcover change | gC/m^2/s |
| **DWT_LIVECROOTN_TO_CWDN** | live coarse root to CWD due to landcover change | gN/m^2/s |
| **DWT_NLOSS** | total nitrogen loss from landcover conversion | gN/m^2/s |
| **DWT_PROD100C_GAIN** | landcover change-driven addition to 100-yr wood product pool | gC/m^2/s |
| **DWT_PROD100N_GAIN** | addition to 100-yr wood product pool | gN/m^2/s |
| **DWT_PROD10C_GAIN** | landcover change-driven addition to 10-yr wood product pool | gC/m^2/s |
| **DWT_PROD10N_GAIN** | addition to 10-yr wood product pool | gN/m^2/s |
| **DWT_SEEDC_TO_DEADSTEM** | seed source to PFT-level deadstem | gC/m^2/s |
| **DWT_SEEDC_TO_LEAF** | seed source to PFT-level leaf | gC/m^2/s |
| **DWT_SEEDN_TO_DEADSTEM** | seed source to PFT-level deadstem | gN/m^2/s |
| **DWT_SEEDN_TO_LEAF** | seed source to PFT-level leaf | gN/m^2/s |
| **EFF_KID** | effective extinction coefficient for indirect from direct | none |
| **EFF_KII** | effective extinction coefficient for indirect from indirect | none |
| **EFF_POROSITY** | effective porosity = porosity - vol_ice | proportion |
| **EFLX_DYNBAL** | dynamic land cover change conversion energy flux | W/m^2 |
| **EFLX_GNET** | net heat flux into ground | W/m^2 |
| **EFLX_LH_TOT** | total latent heat flux [+ to atm] | W/m^2 |
| **EFLX_LH_TOT_R** | Rural total evaporation | W/m^2 |
| **EFLX_LH_TOT_U** | Urban total evaporation | W/m^2 |
| **EFLX_SOIL_GRND** | soil heat flux [+ into soil] | W/m^2 |
| **ELAI** | exposed one-sided leaf area index | m^2/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **EMG** | ground emissivity | proportion |
| **EMV** | vegetation emissivity | proportion |
| **EOPT** | Eopt coefficient for VOC calc | non |
| **ER** | total ecosystem respiration, autotrophic + heterotrophic | gC/m^2/s |
| **ERRH2O** | total water conservation error | mm |
| **ERRH2OSNO** | imbalance in snow depth (liquid water) | mm |
| **ERRSEB** | surface energy conservation error | W/m^2 |
| **ERRSOI** | soil/lake energy conservation error | W/m^2 |
| **ERRSOL** | solar radiation conservation error | W/m^2 |
| **ESAI** | exposed one-sided stem area index | m^2/m^2 |
| **EXCESS_CFLUX** | C flux not allocated due to downregulation | gC/m^2/s |
| **FABD** | flux absorbed by veg per unit direct flux | proportion |
| **FABI** | flux absorbed by veg per unit indirect flux | proportion |
| **FAREA_BURNED** | timestep fractional area burned | proportion |
| **FCEV** | canopy evaporation | W/m^2 |
| **FCOV** | fractional impermeable area | unitless |
| **FCTR** | canopy transpiration | W/m^2 |
| **FDRY** | fraction of foliage that is green and dry | proportion |
| **FGEV** | ground evaporation | W/m^2 |
| **FGR** | heat flux into soil/snow including snow melt | W/m^2 |
| **FGR12** | heat flux between soil layers 1 and 2 | W/m^2 |
| **FGR_R** | Rural heat flux into soil/snow including snow melt | W/m^2 |
| **FGR_U** | Urban heat flux into soil/snow including snow melt | W/m^2 |
| **FIRA** | net infrared (longwave) radiation | W/m^2 |
| **FIRA_R** | Rural net infrared (longwave) radiation | W/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **FIRA_U** | Urban net infrared (longwave) radiation | W/m^2 |
| **FIRE** | emitted infrared (longwave) radiation | W/m^2 |
| **FIRESEASONL** | annual fire season length | days |
| **FIRE_PROB** | daily fire probability | 0-1 |
| **FLDS** | atmospheric longwave radiation | W/m^2 |
| **FPG** | fraction of potential gpp | proportion |
| **FPI** | fraction of potential immobilization | proportion |
| **FPSN** | photosynthesis | umol/m2s |
| **FRAC_ICEOLD** | fraction of ice relative to the tot water | proportion |
| **FROOTC** | fine root C | gC/m^2 |
| **FROOTC_ALLOC** | fine root C allocation | gC/m^2/s |
| **FROOTC_LOSS** | fine root C loss | gC/m^2/s |
| **FROOTC_STORAGE** | fine root C storage | gC/m^2 |
| **FROOTC_STORAGE_TO_XFER** | fine root C shift storage to transfer | gC/m^2/s |
| **FROOTC_TO_LITR1C** | fine root C litterfall to litter 1 C | gC/m^2/s |
| **FROOTC_TO_LITR2C** | fine root C litterfall to litter 2 C | gC/m^2/s |
| **FROOTC_TO_LITR3C** | fine root C litterfall to litter 3 C | gC/m^2/s |
| **FROOTC_TO_LITTER** | fine root C litterfall | gC/m^2/s |
| **FROOTC_XFER** | fine root C transfer | gC/m^2 |
| **FROOTC_XFER_TO_FROOTC** | fine root C growth from storage | gC/m^2/s |
| **FROOTN** | fine root N | gN/m^2 |
| **FROOTN_STORAGE** | fine root N storage | gN/m^2 |
| **FROOTN_STORAGE_TO_XFER** | fine root N shift storage to transfer | gN/m^2/s |
| **FROOTN_TO_LITR1N** | fine root N litterfall to litter 1 N | gN/m^2/s |
| **FROOTN_TO_LITR2N** | fine root N litterfall to litter 2 N | gN/m^2/s |
| **FROOTN_TO_LITR3N** | fine root N litterfall to litter 3 N | gN/m^2/s |
| **FROOTN_TO_LITTER** | fine root N litterfall | gN/m^2/s |
| **FROOTN_XFER** | fine root N transfer | gN/m^2 |
| **FROOTN_XFER_TO_FROOTN** | fine root N growth from storage | gN/m^2/s |
| **FROOT_MR** | fine root maintenance respiration | gC/m^2/s |
| **FSA** | absorbed solar radiation | W/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **FSAT** | fractional area with water table at surface | unitless |
| **FSA_R** | Rural absorbed solar radiation | W/m^2 |
| **FSA_U** | Urban absorbed solar radiation | W/m^2 |
| **FSD24** | direct radiation (last 24hrs) | K |
| **FSD240** | direct radiation (last 240hrs) | K |
| **FSDS** | atmospheric incident solar radiation | W/m^2 |
| **FSDSND** | direct nir incident solar radiation | W/m^2 |
| **FSDSNDLN** | direct nir incident solar radiation at local noon | W/m^2 |
| **FSDSNI** | diffuse nir incident solar radiation | W/m^2 |
| **FSDSVD** | direct vis incident solar radiation | W/m^2 |
| **FSDSVDLN** | direct vis incident solar radiation at local noon | W/m^2 |
| **FSDSVI** | diffuse vis incident solar radiation | W/m^2 |
| **FSH** | sensible heat | W/m^2 |
| **FSH_G** | sensible heat from ground | W/m^2 |
| **FSH_NODYNLNDUSE** | sensible heat not including correction for land use change | W/m^2 |
| **FSH_R** | Rural sensible heat | W/m^2 |
| **FSH_U** | Urban sensible heat | W/m^2 |
| **FSH_V** | sensible heat from veg | W/m^2 |
| **FSI24** | indirect radiation (last 24hrs) | K |
| **FSI240** | indirect radiation (last 240hrs) | K |
| **FSM** | snow melt heat flux | W/m^2 |
| **FSM_R** | Rural snow melt heat flux | W/m^2 |
| **FSM_U** | Urban snow melt heat flux | W/m^2 |
| **FSNO** | fraction of ground covered by snow | unitless |
| **FSR** | reflected solar radiation | W/m^2 |
| **FSRND** | direct nir reflected solar radiation | W/m^2 |
| **FSRNDLN** | direct nir reflected solar radiation at local noon | W/m^2 |
| **FSRNI** | diffuse nir reflected solar radiation | W/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **FSRVD** | direct vis reflected solar radiation | W/m^2 |
| **FSRVDLN** | direct vis reflected solar radiation at local noon | W/m^2 |
| **FSRVI** | diffuse vis reflected solar radiation | W/m^2 |
| **FSUN** | sunlit fraction of canopy | proportion |
| **FSUN24** | fraction sunlit (last 24hrs) | K |
| **FSUN240** | fraction sunlit (last 240hrs) | K |
| **FTDD** | down direct flux below veg per unit dir flx | proportion |
| **FTID** | down indirect flux below veg per unit dir flx | proportion |
| **FTII** | down indirect flux below veg per unit indirect flx | proportion |
| **FV** | friction velocity for dust model | m/s |
| **FWET** | fraction of canopy that is wet | proportion |
| **GAMMA** | total gamma for VOC calc | non |
| **GAMMAA** | gamma A for VOC calc | non |
| **GAMMAC** | gamma C for VOC calc | non |
| **GAMMAL** | gamma L for VOC calc | non |
| **GAMMAP** | gamma P for VOC calc | non |
| **GAMMAS** | gamma S for VOC calc | non |
| **GAMMAT** | gamma T for VOC calc | non |
| **GC_HEAT1** | initial gridcell total heat content | J/m^2 |
| **GC_HEAT2** | post land cover change total heat content | J/m^2 |
| **GC_ICE1** | initial gridcell total ice content | mm |
| **GC_ICE2** | post land cover change total ice content | mm |
| **GC_LIQ1** | initial gridcell total liq content | mm |
| **GC_LIQ2** | post landuse change gridcell total liq content | mm |
| **GDD0** | Growing degree days base 0C from planting | ddays |
| **GDD020** | Twenty year average of growing degree days base 0C from planting | ddays |
| **GDD10** | Growing degree days base 10C from planting | ddays |

| Name | Long-name | Units |
|---|---|---|
| **GDD1020** | Twenty year average of growing degree days base 10C from planting | ddays |
| **GDD8** | Growing degree days base 8C from planting | ddays |
| **GDD820** | Twenty year average of growing degree days base 8C from planting | ddays |
| **GDDHARV** | Growing degree days (gdd) needed to harvest | ddays |
| **GDDPLANT** | Accumulated growing degree days past planting date for crop | ddays |
| **GDDTSOI** | Growing degree-days from planting (top two soil layers) | ddays |
| **GDIR** | leaf projection in solar direction | proportion |
| **GPP** | gross primary production | gC/m^2/s |
| **GR** | total growth respiration | gC/m^2/s |
| **GRESP_STORAGE** | growth respiration storage | gC/m^2 |
| **GRESP_STORAGE_TO_XFER** | growth respiration shift storage to transfer | gC/m^2/s |
| **GRESP_XFER** | growth respiration transfer | gC/m^2 |
| **GROSS_NMIN** | gross rate of N mineralization | gN/m^2/s |
| **H2OCAN** | intercepted water | mm |
| **H2OSNO** | snow depth (liquid water) | mm |
| **H2OSNO_TOP** | mass of snow in top snow layer | kg/m2 |
| **H2OSOI** | volumetric soil water (vegetated landunits only) | mm3/mm3 |
| **HBOT** | canopy bottom | m |
| **HC** | heat content of soil/snow/lake | MJ/m2 |
| **HCSOI** | soil heat content | MJ/m2 |
| **HEAT_FROM_AC** | sensible heat flux put into canyon due to heat removed from air conditioning | W/m^2 |
| **HK** | hydraulic conductivity (vegetated landunits only) | mm/s |
| **HR** | total heterotrophic respiration | gC/m^2/s |
| **HTOP** | canopy top | m |
| **INIT_GPP** | GPP flux before downregulation | gC/m^2/s |
| **LAISHA** | shaded projected leaf area index | none |
| **LAISUN** | sunlit projected leaf area index | none |

| Name | Long-name | Units |
|---|---|---|
| **LAND_UPTAKE** | NEE minus LAND_USE_FLUX, negative for update | gC/m^2/s |
| **LAND_USE_FLUX** | total C emitted from land cover conversion and wood product pools | gC/m^2/s |
| **LEAFC** | leaf C | gC/m^2 |
| **LEAFC_ALLOC** | leaf C allocation | gC/m^2/s |
| **LEAFC_LOSS** | leaf C loss | gC/m^2/s |
| **LEAFC_STORAGE** | leaf C storage | gC/m^2 |
| **LEAFC_STORAGE_TO_XFER** | leaf C shift storage to transfer | gC/m^2/s |
| **LEAFC_TO_LITR1C** | leaf C litterfall to litter 1 C | gC/m^2/s |
| **LEAFC_TO_LITR2C** | leaf C litterfall to litter 2 C | gC/m^2/s |
| **LEAFC_TO_LITR3C** | leaf C litterfall to litter 3 C | gC/m^2/s |
| **LEAFC_TO_LITTER** | leaf C litterfall | gC/m^2/s |
| **LEAFC_XFER** | leaf C transfer | gC/m^2 |
| **LEAFC_XFER_TO_LEAFC** | leaf C growth from storage | gC/m^2/s |
| **LEAFN** | leaf N | gN/m^2 |
| **LEAFN_STORAGE** | leaf N storage | gN/m^2 |
| **LEAFN_STORAGE_TO_XFER** | leaf N shift storage to transfer | gN/m^2/s |
| **LEAFN_TO_LITR1N** | leaf N litterfall to litter 1 N | gN/m^2/s |
| **LEAFN_TO_LITR2N** | leaf N litterfall to litter 2 N | gN/m^2/s |
| **LEAFN_TO_LITR3N** | leaf N litterfall to litter 3 N | gN/m^2/s |
| **LEAFN_TO_LITTER** | leaf N litterfall | gN/m^2/s |
| **LEAFN_TO_RETRANSN** | leaf N to retranslocated N pool | gN/m^2/s |
| **LEAFN_XFER** | leaf N transfer | gN/m^2 |
| **LEAFN_XFER_TO_LEAFN** | leaf N growth from storage | gN/m^2/s |
| **LEAF_MR** | leaf maintenance respiration | gC/m^2/s |
| **LGSF** | long growing season factor | proportion |
| **LITFALL** | litterfall (leaves and fine roots) | gC/m^2/s |
| **LITFIRE** | litter fire losses | gC/m^2/s |
| **LITHR** | litter heterotrophic respiration | gC/m^2/s |
| **LITR1C** | litter labile C | gC/m^2 |
| **LITR1C_TO_SOIL1C** | decomp. of litter 1 C to SOM 1 C | gC/m^2/s |
| **LITR1N** | litter labile N | gN/m^2 |
| **LITR1N_TO_SOIL1N** | decomp. of litter 1 N to SOM 1 N | gN/m^2/s |
| **LITR1_HR** | het. resp. from litter 1 C | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **LITR2C** | litter cellulose C | gC/m^2 |
| **LITR2C_TO_SOIL2C** | decomp. of litter 2 C to SOM 2 C | gC/m^2/s |
| **LITR2N** | litter cellulose N | gN/m^2 |
| **LITR2N_TO_SOIL2N** | decomp. of litter 2 N to SOM 2 N | gN/m^2/s |
| **LITR2_HR** | het. resp. from litter 2 C | gC/m^2/s |
| **LITR3C** | litter lignin C | gC/m^2 |
| **LITR3C_TO_SOIL3C** | decomp. of litter 3 C to SOM 3 C | gC/m^2/s |
| **LITR3N** | litter lignin N | gN/m^2 |
| **LITR3N_TO_SOIL3N** | decomp. of litter 3 N to SOM 3 N | gN/m^2/s |
| **LITR3_HR** | het. resp. from litter 3 C | gC/m^2/s |
| **LITTERC** | litter C | gC/m^2 |
| **LITTERC_HR** | litter C heterotrophic respiration | gC/m^2/s |
| **LITTERC_LOSS** | litter C loss | gC/m^2/s |
| **LIVECROOTC** | live coarse root C | gC/m^2 |
| **LIVECROOTC_STORAGE** | live coarse root C storage | gC/m^2 |
| **LIVECROOTC_STORAGE_TO_XFER** | live coarse root C shift storage to transfer | gC/m^2/s |
| **LIVECROOTC_TO_DEADCROOTC** | live coarse root C turnover | gC/m^2/s |
| **LIVECROOTC_XFER** | live coarse root C transfer | gC/m^2 |
| **LIVECROOTC_XFER_TO_LIVECROOTC** | live coarse root C growth from storage | gC/m^2/s |
| **LIVECROOTN** | live coarse root N | gN/m^2 |
| **LIVECROOTN_STORAGE** | live coarse root N storage | gN/m^2 |
| **LIVECROOTN_STORAGE_TO_XFER** | live coarse root N shift storage to transfer | gN/m^2/s |
| **LIVECROOTN_TO_DEADCROOTN** | live coarse root N turnover | gN/m^2/s |
| **LIVECROOTN_TO_RETRANSN** | live coarse root N to retranslocated N pool | gN/m^2/s |
| **LIVECROOTN_XFER** | live coarse root N transfer | gN/m^2 |
| **LIVECROOTN_XFER_TO_LIVECROOTN** | live coarse root N growth from storage | gN/m^2/s |
| **LIVECROOT_MR** | live coarse root maintenance respiration | gC/m^2/s |
| **LIVESTEMC** | live stem C | gC/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **LIVESTEMC_STORAGE** | live stem C storage | gC/m^2 |
| **LIVESTEMC_STORAGE_TO_XFER** | live stem C shift storage to transfer | gC/m^2/s |
| **LIVESTEMC_TO_DEADSTEMC** | live stem C turnover | gC/m^2/s |
| **LIVESTEMC_XFER** | live stem C transfer | gC/m^2 |
| **LIVESTEMC_XFER_TO_LIVESTEMC** | live stem C growth from storage | gC/m^2/s |
| **LIVESTEMN** | live stem N | gN/m^2 |
| **LIVESTEMN_STORAGE** | live stem N storage | gN/m^2 |
| **LIVESTEMN_STORAGE_TO_XFER** | live stem N shift storage to transfer | gN/m^2/s |
| **LIVESTEMN_TO_DEADSTEMN** | live stem N turnover | gN/m^2/s |
| **LIVESTEMN_TO_RETRANSN** | live stem N to retranslocated N pool | gN/m^2/s |
| **LIVESTEMN_XFER** | live stem N transfer | gN/m^2 |
| **LIVESTEMN_XFER_TO_LIVESTEMN** | live stem N growth from storage | gN/m^2/s |
| **LIVESTEM_MR** | live stem maintenance respiration | gC/m^2/s |
| **LNCSHA** | leaf N concentration per unit projected LAI | gN/m^2 |
| **LNCSUN** | leaf N concentration per unit projected LAI | gN/m^2 |
| **LWdown** | atmospheric longwave radiation | W/m^2 |
| **LWup** | upwelling longwave radiation | W/m^2 |
| **ME** | moisture of extinction | proportion |
| **MEAN_FIRE_PROB** | e-folding mean of daily fire probability | 0-1 |
| **MEG_2met_2s** | MEGAN flux | kg/m2/sec |
| **MEG_2met_nonatriene** | MEGAN flux | kg/m2/sec |
| **MEG_2met_s** | MEGAN flux | kg/m2/sec |
| **MEG_2met_styrene** | MEGAN flux | kg/m2/sec |
| **MEG_3met_3DCTT** | MEGAN flux | kg/m2/sec |
| **MEG_Ehsalate** | MEGAN flux | kg/m2/sec |
| **MEG_MBO_2m3e2ol** | MEGAN flux | kg/m2/sec |
| **MEG_MBO_3m2e1ol** | MEGAN flux | kg/m2/sec |
| **MEG_MBO_3m3e1ol** | MEGAN flux | kg/m2/sec |
| **MEG_Napthalene** | MEGAN flux | kg/m2/sec |

| Name | Long-name | Units |
|---|---|---|
| **MEG_PPPP_2s** | MEGAN flux | kg/m2/sec |
| **MEG_acetaldehyde** | MEGAN flux | kg/m2/sec |
| **MEG_acetic_acid** | MEGAN flux | kg/m2/sec |
| **MEG_acetone** | MEGAN flux | kg/m2/sec |
| **MEG_acoradiene** | MEGAN flux | kg/m2/sec |
| **MEG_ammonia** | MEGAN flux | kg/m2/sec |
| **MEG_anisole** | MEGAN flux | kg/m2/sec |
| **MEG_aromadendrene** | MEGAN flux | kg/m2/sec |
| **MEG_benzaldehyde** | MEGAN flux | kg/m2/sec |
| **MEG_benzyl-acetate** | MEGAN flux | kg/m2/sec |
| **MEG_benzyl-alcohol** | MEGAN flux | kg/m2/sec |
| **MEG_bergamotene_a** | MEGAN flux | kg/m2/sec |
| **MEG_bergamotene_b** | MEGAN flux | kg/m2/sec |
| **MEG_bisabolene_a** | MEGAN flux | kg/m2/sec |
| **MEG_bisabolene_b** | MEGAN flux | kg/m2/sec |
| **MEG_bornene** | MEGAN flux | kg/m2/sec |
| **MEG_borneol** | MEGAN flux | kg/m2/sec |
| **MEG_bornyl_ACT** | MEGAN flux | kg/m2/sec |
| **MEG_bourbonene_b** | MEGAN flux | kg/m2/sec |
| **MEG_butanone_2** | MEGAN flux | kg/m2/sec |
| **MEG_butene** | MEGAN flux | kg/m2/sec |
| **MEG_cadinene_d** | MEGAN flux | kg/m2/sec |
| **MEG_cadinene_g** | MEGAN flux | kg/m2/sec |
| **MEG_camphene** | MEGAN flux | kg/m2/sec |
| **MEG_camphor** | MEGAN flux | kg/m2/sec |
| **MEG_carbon_2s** | MEGAN flux | kg/m2/sec |
| **MEG_carbon_monoxide** | MEGAN flux | kg/m2/sec |
| **MEG_carbonyl_s** | MEGAN flux | kg/m2/sec |
| **MEG_carene_3** | MEGAN flux | kg/m2/sec |
| **MEG_caryophyllene_b** | MEGAN flux | kg/m2/sec |
| **MEG_cedrene_a** | MEGAN flux | kg/m2/sec |
| **MEG_cedrol** | MEGAN flux | kg/m2/sec |
| **MEG_cineole_1_8** | MEGAN flux | kg/m2/sec |
| **MEG_copaene_a** | MEGAN flux | kg/m2/sec |
| **MEG_cubebene_a** | MEGAN flux | kg/m2/sec |
| **MEG_cubebene_b** | MEGAN flux | kg/m2/sec |
| **MEG_cymene_o** | MEGAN flux | kg/m2/sec |
| **MEG_cymene_p** | MEGAN flux | kg/m2/sec |

| Name | Long-name | Units |
|---|---|---|
| **MEG_decanal** | MEGAN flux | kg/m2/sec |
| **MEG_diallyl_2s** | MEGAN flux | kg/m2/sec |
| **MEG_dodecene_1** | MEGAN flux | kg/m2/sec |
| **MEG_elemene_b** | MEGAN flux | kg/m2/sec |
| **MEG_estragole** | MEGAN flux | kg/m2/sec |
| **MEG_ethane** | MEGAN flux | kg/m2/sec |
| **MEG_ethanol** | MEGAN flux | kg/m2/sec |
| **MEG_ethene** | MEGAN flux | kg/m2/sec |
| **MEG_farnescene_a** | MEGAN flux | kg/m2/sec |
| **MEG_farnescene_b** | MEGAN flux | kg/m2/sec |
| **MEG_fenchene_a** | MEGAN flux | kg/m2/sec |
| **MEG_fenchone** | MEGAN flux | kg/m2/sec |
| **MEG_formaldehyde** | MEGAN flux | kg/m2/sec |
| **MEG_formic_acid** | MEGAN flux | kg/m2/sec |
| **MEG_geranyl_acetone** | MEGAN flux | kg/m2/sec |
| **MEG_germacrene_B** | MEGAN flux | kg/m2/sec |
| **MEG_germacrene_D** | MEGAN flux | kg/m2/sec |
| **MEG_gurjunene_b** | MEGAN flux | kg/m2/sec |
| **MEG_heptanal** | MEGAN flux | kg/m2/sec |
| **MEG_heptane** | MEGAN flux | kg/m2/sec |
| **MEG_heptanone** | MEGAN flux | kg/m2/sec |
| **MEG_hexanal** | MEGAN flux | kg/m2/sec |
| **MEG_hexane** | MEGAN flux | kg/m2/sec |
| **MEG_hexanol_1** | MEGAN flux | kg/m2/sec |
| **MEG_hexenal_c3** | MEGAN flux | kg/m2/sec |
| **MEG_hexenal_t2** | MEGAN flux | kg/m2/sec |
| **MEG_hexenol_c3** | MEGAN flux | kg/m2/sec |
| **MEG_hexenyl_ACT_c3** | MEGAN flux | kg/m2/sec |
| **MEG_homosalate** | MEGAN flux | kg/m2/sec |
| **MEG_humulene_a** | MEGAN flux | kg/m2/sec |
| **MEG_humulene_g** | MEGAN flux | kg/m2/sec |
| **MEG_hydrogen_cyanide** | MEGAN flux | kg/m2/sec |
| **MEG_hydrogen_s** | MEGAN flux | kg/m2/sec |
| **MEG_indole** | MEGAN flux | kg/m2/sec |
| **MEG_ionone_b** | MEGAN flux | kg/m2/sec |
| **MEG_ipsenol** | MEGAN flux | kg/m2/sec |
| **MEG_isolongifolene** | MEGAN flux | kg/m2/sec |
| **MEG_isoprene** | MEGAN flux | kg/m2/sec |

| Name | Long-name | Units |
|---|---|---|
| **MEG_jasmone** | MEGAN flux | kg/m2/sec |
| **MEG_limonene** | MEGAN flux | kg/m2/sec |
| **MEG_linalool** | MEGAN flux | kg/m2/sec |
| **MEG_linalool_OXD_c** | MEGAN flux | kg/m2/sec |
| **MEG_linalool_OXD_t** | MEGAN flux | kg/m2/sec |
| **MEG_longifolene** | MEGAN flux | kg/m2/sec |
| **MEG_longipinene** | MEGAN flux | kg/m2/sec |
| **MEG_met_benzoate** | MEGAN flux | kg/m2/sec |
| **MEG_met_bromide** | MEGAN flux | kg/m2/sec |
| **MEG_met_chloride** | MEGAN flux | kg/m2/sec |
| **MEG_met_heptenone** | MEGAN flux | kg/m2/sec |
| **MEG_met_iodide** | MEGAN flux | kg/m2/sec |
| **MEG_met_jasmonate** | MEGAN flux | kg/m2/sec |
| **MEG_met_mercaptan** | MEGAN flux | kg/m2/sec |
| **MEG_met_propenyl_2s** | MEGAN flux | kg/m2/sec |
| **MEG_met_salicylate** | MEGAN flux | kg/m2/sec |
| **MEG_meta-cymenene** | MEGAN flux | kg/m2/sec |
| **MEG_methane** | MEGAN flux | kg/m2/sec |
| **MEG_methanol** | MEGAN flux | kg/m2/sec |
| **MEG_muurolene_a** | MEGAN flux | kg/m2/sec |
| **MEG_muurolene_g** | MEGAN flux | kg/m2/sec |
| **MEG_myrcene** | MEGAN flux | kg/m2/sec |
| **MEG_myrtenal** | MEGAN flux | kg/m2/sec |
| **MEG_nerolidol_c** | MEGAN flux | kg/m2/sec |
| **MEG_nerolidol_t** | MEGAN flux | kg/m2/sec |
| **MEG_neryl_acetone** | MEGAN flux | kg/m2/sec |
| **MEG_nitric_OXD** | MEGAN flux | kg/m2/sec |
| **MEG_nitrous_OXD** | MEGAN flux | kg/m2/sec |
| **MEG_nonanal** | MEGAN flux | kg/m2/sec |
| **MEG_nonenal** | MEGAN flux | kg/m2/sec |
| **MEG_ocimene_al** | MEGAN flux | kg/m2/sec |
| **MEG_ocimene_c_b** | MEGAN flux | kg/m2/sec |
| **MEG_ocimene_t_b** | MEGAN flux | kg/m2/sec |
| **MEG_octanal** | MEGAN flux | kg/m2/sec |
| **MEG_octanol** | MEGAN flux | kg/m2/sec |
| **MEG_octenol_1e3ol** | MEGAN flux | kg/m2/sec |
| **MEG_oxopentanal** | MEGAN flux | kg/m2/sec |
| **MEG_pentanal** | MEGAN flux | kg/m2/sec |

| Name | Long-name | Units |
|---|---|---|
| **MEG_pentane** | MEGAN flux | kg/m2/sec |
| **MEG_phellandrene_a** | MEGAN flux | kg/m2/sec |
| **MEG_phellandrene_b** | MEGAN flux | kg/m2/sec |
| **MEG_phenyl_CCO** | MEGAN flux | kg/m2/sec |
| **MEG_pinene_a** | MEGAN flux | kg/m2/sec |
| **MEG_pinene_b** | MEGAN flux | kg/m2/sec |
| **MEG_piperitone** | MEGAN flux | kg/m2/sec |
| **MEG_propane** | MEGAN flux | kg/m2/sec |
| **MEG_propene** | MEGAN flux | kg/m2/sec |
| **MEG_pyruvic_acid** | MEGAN flux | kg/m2/sec |
| **MEG_sabinene** | MEGAN flux | kg/m2/sec |
| **MEG_selinene_b** | MEGAN flux | kg/m2/sec |
| **MEG_selinene_d** | MEGAN flux | kg/m2/sec |
| **MEG_terpinene_a** | MEGAN flux | kg/m2/sec |
| **MEG_terpinene_g** | MEGAN flux | kg/m2/sec |
| **MEG_terpineol_4** | MEGAN flux | kg/m2/sec |
| **MEG_terpineol_a** | MEGAN flux | kg/m2/sec |
| **MEG_terpinolene** | MEGAN flux | kg/m2/sec |
| **MEG_terpinyl_ACT_a** | MEGAN flux | kg/m2/sec |
| **MEG_tetradecene_1** | MEGAN flux | kg/m2/sec |
| **MEG_thujene_a** | MEGAN flux | kg/m2/sec |
| **MEG_thujone_a** | MEGAN flux | kg/m2/sec |
| **MEG_thujone_b** | MEGAN flux | kg/m2/sec |
| **MEG_toluene** | MEGAN flux | kg/m2/sec |
| **MEG_tricyclene** | MEGAN flux | kg/m2/sec |
| **MEG_verbenene** | MEGAN flux | kg/m2/sec |
| **MR** | maintenance respiration | gC/m^2/s |
| **M_CWDC_TO_FIRE** | coarse woody debris C fire loss | gC/m^2/s |
| **M_CWDN_TO_FIRE** | coarse woody debris N fire loss | gN/m^2/s |
| **M_DEADCROOTC_STORAGE_TO_FIRE** | dead coarse root C storage fire loss | gC/m^2/s |
| **M_DEADCROOTC_STORAGE_TO_LITR1C** | dead coarse root C storage mortality to litter 1 C | gC/m^2/s |
| **M_DEADCROOTC_STORAGE_TO_LITTER** | dead coarse root C storage mortality | gC/m^2/s |
| **M_DEADCROOTC_TO_CWDC** | dead coarse root C mortality to coarse woody debris C | gC/m^2/s |
| **M_DEADCROOTC_TO_CWDC_FIRE** | dead coarse root C to to woody debris C by fire | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_DEADCROOTC_TO_FIRE** | dead coarse root C fire loss | gC/m^2/s |
| **M_DEADCROOTC_TO_LITTER** | dead coarse root C mortality | gC/m^2/s |
| **M_DEADCROOTC_TO_LITTER_FIRE** | dead coarse root C fire mortality to litter | gC/m^2/s |
| **M_DEADCROOTC_XFER_TO_FIRE** | dead coarse root C transfer fire loss | gC/m^2/s |
| **M_DEADCROOTC_XFER_TO_LITR1C** | dead coarse root C transfer mortality to litter 1 C | gC/m^2/s |
| **M_DEADCROOTC_XFER_TO_LITTER** | dead coarse root C transfer mortality | gC/m^2/s |
| **M_DEADCROOTN_STORAGE_TO_FIRE** | dead coarse root N storage fire loss | gN/m^2/s |
| **M_DEADCROOTN_STORAGE_TO_LITR1N** | dead coarse root N storage mortality to litter 1 N | gN/m^2/s |
| **M_DEADCROOTN_STORAGE_TO_LITTER** | dead coarse root N storage mortality | gN/m^2/s |
| **M_DEADCROOTN_TO_CWDN** | dead coarse root N mortality to coarse woody debris N | gN/m^2/s |
| **M_DEADCROOTN_TO_CWDN_FIRE** | dead coarse root N to to woody debris N by fire | gN/m^2/s |
| **M_DEADCROOTN_TO_FIRE** | dead coarse root N fire loss | gN/m^2/s |
| **M_DEADCROOTN_TO_LITTER** | dead coarse root N mortality | gN/m^2/s |
| **M_DEADCROOTN_TO_LITTER_FIRE** | dead coarse root N fire mortality to litter | gN/m^2/s |
| **M_DEADCROOTN_XFER_TO_FIRE** | dead coarse root N transfer fire loss | gN/m^2/s |
| **M_DEADCROOTN_XFER_TO_LITR1N** | dead coarse root N transfer mortality to litter 1 N | gN/m^2/s |
| **M_DEADCROOTN_XFER_TO_LITTER** | dead coarse root N transfer mortality | gN/m^2/s |
| **M_DEADSTEMC_STORAGE_TO_FIRE** | dead stem C storage fire loss | gC/m^2/s |
| **M_DEADSTEMC_STORAGE_TO_LITR1C** | dead stem C storage mortality to litter 1 C | gC/m^2/s |
| **M_DEADSTEMC_STORAGE_TO_LITTER** | dead stem C storage mortality | gC/m^2/s |
| **M_DEADSTEMC_TO_CWDC** | dead stem C mortality to coarse woody debris C | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_DEADSTEMC_TO_CWDC_FIRE** | dead stem C to coarse woody debris C by fire | gC/m^2/s |
| **M_DEADSTEMC_TO_FIRE** | dead stem C fire loss | gC/m^2/s |
| **M_DEADSTEMC_TO_LITTER** | dead stem C mortality | gC/m^2/s |
| **M_DEADSTEMC_TO_LITTER_FIRE** | dead stem C fire mortality to litter | gC/m^2/s |
| **M_DEADSTEMC_XFER_TO_FIRE** | dead stem C transfer fire loss | gC/m^2/s |
| **M_DEADSTEMC_XFER_TO_LITR1C** | dead stem C transfer mortality to litter 1 C | gC/m^2/s |
| **M_DEADSTEMC_XFER_TO_LITTER** | dead stem C transfer mortality | gC/m^2/s |
| **M_DEADSTEMN_STORAGE_TO_FIRE** | dead stem N storage fire loss | gN/m^2/s |
| **M_DEADSTEMN_STORAGE_TO_LITR1N** | dead stem N storage mortality to litter 1 N | gN/m^2/s |
| **M_DEADSTEMN_STORAGE_TO_LITTER** | dead stem N storage mortality | gN/m^2/s |
| **M_DEADSTEMN_TO_CWDN** | dead stem N mortality to coarse woody debris N | gN/m^2/s |
| **M_DEADSTEMN_TO_CWDN_FIRE** | dead stem N to coarse woody debris N by fire | gN/m^2/s |
| **M_DEADSTEMN_TO_FIRE** | dead stem N fire loss | gN/m^2/s |
| **M_DEADSTEMN_TO_LITTER** | dead stem N mortality | gN/m^2/s |
| **M_DEADSTEMN_TO_LITTER_FIRE** | dead stem N fire mortality to litter | gN/m^2/s |
| **M_DEADSTEMN_XFER_TO_FIRE** | dead stem N transfer fire loss | gN/m^2/s |
| **M_DEADSTEMN_XFER_TO_LITR1N** | dead stem N transfer mortality to litter 1 N | gN/m^2/s |
| **M_DEADSTEMN_XFER_TO_LITTER** | dead stem N transfer mortality | gN/m^2/s |
| **M_FROOTC_STORAGE_TO_FIRE** | fine root C storage fire loss | gC/m^2/s |
| **M_FROOTC_STORAGE_TO_LITR1C** | fine root C storage mortality to litter 1 C | gC/m^2/s |
| **M_FROOTC_STORAGE_TO_LITTER** | fine root C storage mortality | gC/m^2/s |
| **M_FROOTC_TO_FIRE** | fine root C fire loss | gC/m^2/s |
| **M_FROOTC_TO_LITR1C** | fine root C mortality to litter 1 C | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_FROOTC_TO_LITR2C** | fine root C mortality to litter 2 C | gC/m^2/s |
| **M_FROOTC_TO_LITR3C** | fine root C mortality to litter 3 C | gC/m^2/s |
| **M_FROOTC_TO_LITTER** | fine root C mortality | gC/m^2/s |
| **M_FROOTC_XFER_TO_FIRE** | fine root C transfer fire loss | gC/m^2/s |
| **M_FROOTC_XFER_TO_LITR1C** | fine root C transfer mortality to litter 1 C | gC/m^2/s |
| **M_FROOTC_XFER_TO_LITTER** | fine root C transfer mortality | gC/m^2/s |
| **M_FROOTN_STORAGE_TO_FIRE** | fine root N storage fire loss | gN/m^2/s |
| **M_FROOTN_STORAGE_TO_LITR1N** | fine root N storage mortality to litter 1 N | gN/m^2/s |
| **M_FROOTN_STORAGE_TO_LITTER** | fine root N storage mortality | gN/m^2/s |
| **M_FROOTN_TO_FIRE** | fine root N fire loss | gN/m^2/s |
| **M_FROOTN_TO_LITR1N** | fine root N mortality to litter 1 N | gN/m^2/s |
| **M_FROOTN_TO_LITR2N** | fine root N mortality to litter 2 N | gN/m^2/s |
| **M_FROOTN_TO_LITR3N** | fine root N mortality to litter 3 N | gN/m^2/s |
| **M_FROOTN_TO_LITTER** | fine root N mortality | gN/m^2/s |
| **M_FROOTN_XFER_TO_FIRE** | fine root N transfer fire loss | gN/m^2/s |
| **M_FROOTN_XFER_TO_LITR1N** | fine root N transfer mortality to litter 1 N | gN/m^2/s |
| **M_FROOTN_XFER_TO_LITTER** | fine root N transfer mortality | gN/m^2/s |
| **M_GRESP_STORAGE_TO_FIRE** | growth respiration storage fire loss | gC/m^2/s |
| **M_GRESP_STORAGE_TO_LITR1C** | growth respiration storage mortality to litter 1 C | gC/m^2/s |
| **M_GRESP_STORAGE_TO_LITTER** | growth respiration storage mortality | gC/m^2/s |
| **M_GRESP_XFER_TO_FIRE** | growth respiration transfer fire loss | gC/m^2/s |
| **M_GRESP_XFER_TO_LITR1C** | growth respiration transfer mortality to litter 1 C | gC/m^2/s |
| **M_GRESP_XFER_TO_LITTER** | growth respiration transfer mortality | gC/m^2/s |
| **M_LEAFC_STORAGE_TO_FIRE** | leaf C storage fire loss | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_LEAFC_STORAGE_TO_LITR1C** | leaf C storage mortality to litter 1 C | gC/m^2/s |
| **M_LEAFC_STORAGE_TO_LITTER** | leaf C storage mortality | gC/m^2/s |
| **M_LEAFC_TO_FIRE** | leaf C fire loss | gC/m^2/s |
| **M_LEAFC_TO_LITR1C** | leaf C mortality to litter 1 C | gC/m^2/s |
| **M_LEAFC_TO_LITR2C** | leaf C mortality to litter 2 C | gC/m^2/s |
| **M_LEAFC_TO_LITR3C** | leaf C mortality to litter 3 C | gC/m^2/s |
| **M_LEAFC_TO_LITTER** | leaf C mortality | gC/m^2/s |
| **M_LEAFC_XFER_TO_FIRE** | leaf C transfer fire loss | gC/m^2/s |
| **M_LEAFC_XFER_TO_LITR1C** | leaf C transfer mortality to litter 1 C | gC/m^2/s |
| **M_LEAFC_XFER_TO_LITTER** | leaf C transfer mortality | gC/m^2/s |
| **M_LEAFN_STORAGE_TO_FIRE** | leaf N storage fire loss | gN/m^2/s |
| **M_LEAFN_STORAGE_TO_LITR1N** | leaf N storage mortality to litter 1 N | gN/m^2/s |
| **M_LEAFN_STORAGE_TO_LITTER** | leaf N storage mortality | gN/m^2/s |
| **M_LEAFN_TO_FIRE** | leaf N fire loss | gN/m^2/s |
| **M_LEAFN_TO_LITR1N** | leaf N mortality to litter 1 N | gN/m^2/s |
| **M_LEAFN_TO_LITR2N** | leaf N mortality to litter 2 N | gN/m^2/s |
| **M_LEAFN_TO_LITR3N** | leaf N mortality to litter 3 N | gN/m^2/s |
| **M_LEAFN_TO_LITTER** | leaf N mortality | gN/m^2/s |
| **M_LEAFN_XFER_TO_FIRE** | leaf N transfer fire loss | gN/m^2/s |
| **M_LEAFN_XFER_TO_LITR1N** | leaf N transfer mortality to litter 1 N | gN/m^2/s |
| **M_LEAFN_XFER_TO_LITTER** | leaf N transfer mortality | gN/m^2/s |
| **M_LITR1C_TO_FIRE** | litter 1 C fire loss | gC/m^2/s |
| **M_LITR1N_TO_FIRE** | litter 1 N fire loss | gN/m^2/s |
| **M_LITR2C_TO_FIRE** | litter 2 C fire loss | gC/m^2/s |
| **M_LITR2N_TO_FIRE** | litter 2 N fire loss | gN/m^2/s |
| **M_LITR3C_TO_FIRE** | litter 3 C fire loss | gC/m^2/s |
| **M_LITR3N_TO_FIRE** | litter 3 N fire loss | gN/m^2/s |
| **M_LIVECROOTC_STORAGE_TO_FIRE** | live coarse root C storage fire loss | gC/m^2/s |
| **M_LIVECROOTC_STORAGE_TO_LITR1C** | live coarse root C storage mortality to litter 1 C | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_LIVECROOTC_STORAGE_TO_LITTER** | live coarse root C storage mortality | gC/m^2/s |
| **M_LIVECROOTC_TO_CWDC** | live coarse root C mortality to coarse woody debris C | gC/m^2/s |
| **M_LIVECROOTC_TO_FIRE** | live coarse root C fire loss | gC/m^2/s |
| **M_LIVECROOTC_TO_LITTER** | live coarse root C mortality | gC/m^2/s |
| **M_LIVECROOTC_XFER_TO_FIRE** | live coarse root C transfer fire loss | gC/m^2/s |
| **M_LIVECROOTC_XFER_TO_LITR1C** | live coarse root C transfer mortality to litter 1 C | gC/m^2/s |
| **M_LIVECROOTC_XFER_TO_LITTER** | live coarse root C transfer mortality | gC/m^2/s |
| **M_LIVECROOTN_STORAGE_TO_FIRE** | live coarse root N storage fire loss | gN/m^2/s |
| **M_LIVECROOTN_STORAGE_TO_LITR1N** | live coarse root N storage mortality to litter 1 N | gN/m^2/s |
| **M_LIVECROOTN_STORAGE_TO_LITTER** | live coarse root N storage mortality | gN/m^2/s |
| **M_LIVECROOTN_TO_CWDN** | live coarse root N mortality to coarse woody debris N | gN/m^2/s |
| **M_LIVECROOTN_TO_FIRE** | live coarse root N fire loss | gN/m^2/s |
| **M_LIVECROOTN_TO_LITTER** | live coarse root N mortality | gN/m^2/s |
| **M_LIVECROOTN_XFER_TO_FIRE** | live coarse root N transfer fire loss | gN/m^2/s |
| **M_LIVECROOTN_XFER_TO_LITR1N** | live coarse root N transfer mortality to litter 1 N | gN/m^2/s |
| **M_LIVECROOTN_XFER_TO_LITTER** | live coarse root N transfer mortality | gN/m^2/s |
| **M_LIVESTEMC_STORAGE_TO_FIRE** | live stem C storage fire loss | gC/m^2/s |
| **M_LIVESTEMC_STORAGE_TO_LITR1C** | live stem C storage mortality to litter 1 C | gC/m^2/s |
| **M_LIVESTEMC_STORAGE_TO_LITTER** | live stem C storage mortality | gC/m^2/s |
| **M_LIVESTEMC_TO_CWDC** | live stem C mortality to coarse woody debris C | gC/m^2/s |
| **M_LIVESTEMC_TO_FIRE** | live stem C fire loss | gC/m^2/s |
| **M_LIVESTEMC_TO_LITTER** | live stem C mortality | gC/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **M_LIVESTEMC_XFER_TO_FIRE** | live stem C transfer fire loss | gC/m^2/s |
| **M_LIVESTEMC_XFER_TO_LITR1C** | live stem C transfer mortality to litter 1 C | gC/m^2/s |
| **M_LIVESTEMC_XFER_TO_LITTER** | live stem C transfer mortality | gC/m^2/s |
| **M_LIVESTEMN_STORAGE_TO_FIRE** | live stem N storage fire loss | gN/m^2/s |
| **M_LIVESTEMN_STORAGE_TO_LITR1N** | live stem N storage mortality to litter 1 N | gN/m^2/s |
| **M_LIVESTEMN_STORAGE_TO_LITTER** | live stem N storage mortality | gN/m^2/s |
| **M_LIVESTEMN_TO_CWDN** | live stem N mortality to coarse woody debris N | gN/m^2/s |
| **M_LIVESTEMN_TO_FIRE** | live stem N fire loss | gN/m^2/s |
| **M_LIVESTEMN_TO_LITTER** | live stem N mortality | gN/m^2/s |
| **M_LIVESTEMN_XFER_TO_FIRE** | live stem N transfer fire loss | gN/m^2/s |
| **M_LIVESTEMN_XFER_TO_LITR1N** | live stem N transfer mortality to litter 1 N | gN/m^2/s |
| **M_LIVESTEMN_XFER_TO_LITTER** | live stem N transfer mortality | gN/m^2/s |
| **M_RETRANSN_TO_FIRE** | retranslocated N pool fire loss | gN/m^2/s |
| **M_RETRANSN_TO_LITR1N** | retranslocated N pool mortality to litter 1 N | gN/m^2/s |
| **M_RETRANSN_TO_LITTER** | retranslocated N pool mortality | gN/m^2/s |
| **NBP** | net biome production, includes fire, landuse, and harvest flux, positive for sink | gC/m^2/s |
| **NDEPLOY** | total N deployed in new growth | gN/m^2/s |
| **NDEP_TO_SMINN** | atmospheric N deposition to soil mineral N | gN/m^2/s |
| **NEE** | net ecosystem exchange of carbon, includes fire, landuse, harvest, and hrv_xsmrpool flux, positive for source | gC/m^2/s |
| **NEP** | net ecosystem production, excludes fire, landuse, and harvest flux, positive for sink | gC/m^2/s |
| **NET_NMIN** | net rate of N mineralization | gN/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **NFIX_TO_SMINN** | symbiotic/asymbiotic N fixation to soil mineral N | gN/m^2/s |
| **NPOOL** | temporary plant N pool | gN/m^2 |
| **NPOOL_TO_DEADCROOTN** | allocation to dead coarse root N | gN/m^2/s |
| **NPOOL_TO_DEADCROOTN_STORAGE** | allocation to dead coarse root N storage | gN/m^2/s |
| **NPOOL_TO_DEADSTEMN** | allocation to dead stem N | gN/m^2/s |
| **NPOOL_TO_DEADSTEMN_STORAGE** | allocation to dead stem N storage | gN/m^2/s |
| **NPOOL_TO_FROOTN** | allocation to fine root N | gN/m^2/s |
| **NPOOL_TO_FROOTN_STORAGE** | allocation to fine root N storage | gN/m^2/s |
| **NPOOL_TO_LEAFN** | allocation to leaf N | gN/m^2/s |
| **NPOOL_TO_LEAFN_STORAGE** | allocation to leaf N storage | gN/m^2/s |
| **NPOOL_TO_LIVECROOTN** | allocation to live coarse root N | gN/m^2/s |
| **NPOOL_TO_LIVECROOTN_STORAGE** | allocation to live coarse root N storage | gN/m^2/s |
| **NPOOL_TO_LIVESTEMN** | allocation to live stem N | gN/m^2/s |
| **NPOOL_TO_LIVESTEMN_STORAGE** | allocation to live stem N storage | gN/m^2/s |
| **NPP** | net primary production | gC/m^2/s |
| **N_ALLOMETRY** | N allocation index | none |
| **OCDEP** | total OC deposition (dry+wet) from atmosphere | kg/m^2/s |
| **OFFSET_COUNTER** | offset days counter | days |
| **OFFSET_FDD** | offset freezing degree days counter | C degree-days |
| **OFFSET_FLAG** | offset flag | none |
| **OFFSET_SWI** | offset soil water index | none |
| **OMEGA** | fraction of intercepted radiation that is scattered | proportion |
| **ONSET_COUNTER** | onset days counter | days |
| **ONSET_FDD** | onset freezing degree days counter | C degree-days |
| **ONSET_FLAG** | onset flag | none |
| **ONSET_GDD** | onset growing degree days | C degree-days |
| **ONSET_GDDFLAG** | onset flag for growing degree day sum | none |
| **ONSET_SWI** | onset soil water index | none |

| Name | Long-name | Units |
|---|---|---|
| **PAR240_shade** | shade PAR (240 hrs) | umol/m2/s |
| **PAR240_sun** | sunlit PAR (240 hrs) | umol/m2/s |
| **PAR24_shade** | shade PAR (24 hrs) | umol/m2/s |
| **PAR24_sun** | sunlit PAR (24 hrs) | umol/m2/s |
| **PARSHA** | average absorbed PAR for shaded leaves | W/m^2 |
| **PARSUN** | average absorbed PAR for sunlit leaves | W/m^2 |
| **PAR_shade** | shade PAR | umol/m2/s |
| **PAR_sun** | sunlit PAR | umol/m2/s |
| **PBOT** | atmospheric pressure | Pa |
| **PCO2** | atmospheric partial pressure of CO2 | Pa |
| **PFT_CTRUNC** | pft-level sink for C truncation | gC/m^2 |
| **PFT_FIRE_CLOSS** | total pft-level fire C loss | gC/m^2/s |
| **PFT_FIRE_NLOSS** | total pft-level fire N loss | gN/m^2/s |
| **PFT_NTRUNC** | pft-level sink for N truncation | gN/m^2 |
| **PLANT_CALLOC** | total allocated C flux | gC/m^2/s |
| **PLANT_NALLOC** | total allocated N flux | gN/m^2/s |
| **PLANT_NDEMAND** | N flux required to support initial GPP | gN/m^2/s |
| **POTENTIAL_IMMOB** | potential N immobilization | gN/m^2/s |
| **PREV_DAYL** | daylength from previous timestep | s |
| **PREV_FROOTC_TO_LITTER** | previous timestep froot C litterfall flux | gC/m^2/s |
| **PREV_LEAFC_TO_LITTER** | previous timestep leaf C litterfall flux | gC/m^2/s |
| **PROD100C** | 100-yr wood product C | gC/m^2 |
| **PROD100C_LOSS** | loss from 100-yr wood product pool | gC/m^2/s |
| **PROD100N** | 100-yr wood product N | gN/m^2 |
| **PROD100N_LOSS** | loss from 100-yr wood product pool | gN/m^2/s |
| **PROD10C** | 10-yr wood product C | gC/m^2 |
| **PROD10C_LOSS** | loss from 10-yr wood product pool | gC/m^2/s |
| **PROD10N** | 10-yr wood product N | gN/m^2 |
| **PROD10N_LOSS** | loss from 10-yr wood product pool | gN/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **PRODUCT_CLOSS** | total carbon loss from wood product pools | gC/m^2/s |
| **PRODUCT_NLOSS** | total N loss from wood product pools | gN/m^2/s |
| **PSNSHA** | shaded leaf photosynthesis | umolCO2/m^2/s |
| **PSNSHADE_TO_CPOOL** | C fixation from shaded canopy | gC/m^2/s |
| **PSNSUN** | sunlit leaf photosynthesis | umolCO2/m^2/s |
| **PSNSUN_TO_CPOOL** | C fixation from sunlit canopy | gC/m^2/s |
| **PSurf** | surface pressure | Pa |
| **Q2M** | 2m specific humidity | kg/kg |
| **QBOT** | atmospheric specific humidity | kg/kg |
| **QCHARGE** | aquifer recharge rate (vegetated landunits only) | mm/s |
| **QDRAI** | sub-surface drainage | mm/s |
| **QDRIP** | throughfall | mm/s |
| **QFLX_DEW_GRND** | ground surface dew formation | mm H2O/s |
| **QFLX_DEW_SNOW** | surface dew added to snow pacK | mm H2O/s |
| **QFLX_EVAP_GRND** | ground surface evaporation | mm H2O/s |
| **QFLX_EVAP_TOT** | qflx_evap_soi + qflx_evap_can + qflx_tran_veg | mm H2O/s |
| **QFLX_EVAP_VEG** | vegetation evaporation | mm H2O/s |
| **QFLX_ICE_DYNBAL** | ice dynamic land cover change conversion runoff flux | mm/s |
| **QFLX_LIQ_DYNBAL** | liq dynamic land cover change conversion runoff flux | mm/s |
| **QFLX_RAIN_GRND** | rain on ground after interception | mm H2O/s |
| **QFLX_SNOW_GRND** | snow on ground after interception | mm H2O/s |
| **QFLX_SUB_SNOW** | sublimation rate from snow pack | mm H2O/s |
| **QICE** | ice growth/melt | mm/s |
| **QICE_FRZ** | ice growth | mm/s |
| **QICE_MELT** | ice melt | mm/s |
| **QINFL** | infiltration | mm/s |
| **QINTR** | interception | mm/s |
| **QIRRIG** | water added through irrigation | mm/s |
| **QOVER** | surface runoff | mm/s |
| **QRGWL** | surface runoff at glaciers (liquid only), wetlands, lakes | mm/s |
| **QRUNOFF** | total liquid runoff (does not include QSNWCPICE) | mm/s |

| Name | Long-name | Units |
|---|---|---|
| **QRUNOFF_NODYNLNDUSE** | total liquid runoff (does not include QSNWCPICE) not including correction for land use change | mm/s |
| **QRUNOFF_R** | Rural total runoff | mm/s |
| **QRUNOFF_U** | Urban total runoff | mm/s |
| **QSNOFRZ** | column-integrated snow freezing rate | kg/m2/s |
| **QSNOMELT** | snow melt | mm/s |
| **QSNWCPICE** | excess snowfall due to snow capping | mm/s |
| **QSNWCPICE_NODYNLNDUSE** | excess snowfall due to snow capping not including correction for land use change | mm H2O/s |
| **QSNWCPLIQ** | excess rainfall due to snow capping | mm H2O/s |
| **QSOIL** | Ground evaporation (soil/snow evaporation + soil/snow sublimation - dew) | mm/s |
| **QTOPSOIL** | water input to surface | mm/s |
| **QVEGE** | canopy evaporation | mm/s |
| **QVEGT** | canopy transpiration | mm/s |
| **Qair** | atmospheric specific humidity | kg/kg |
| **Qanth** | anthropogenic heat flux | W/m^2 |
| **Qh** | sensible heat | W/m^2 |
| **Qle** | total evaporation | W/m^2 |
| **Qstor** | storage heat flux (includes snowmelt) | W/m^2 |
| **Qtau** | momentum flux | kg/m/s^2 |
| **RAIN** | atmospheric rain | mm/s |
| **RAM1** | aerodynamical resistance | s/m |
| **RC13_CANAIR** | C13/C(12+13) for canopy air | proportion |
| **RC13_PSNSHA** | C13/C(12+13) for shaded photosynthesis | proportion |
| **RC13_PSNSUN** | C13/C(12+13) for sunlit photosynthesis | proportion |
| **RETRANSN** | plant pool of retranslocated N | gN/m^2 |
| **RETRANSN_TO_NPOOL** | deployment of retranslocated N | gN/m^2/s |
| **RH** | atmospheric relative humidity | % |
| **RH2M** | 2m relative humidity | % |
| **RH2M_R** | Rural 2m specific humidity | % |

| Name | Long-name | Units |
|---|---|---|
| **RH2M_U** | Urban 2m relative humidity | % |
| **ROOTFR** | fraction of roots in each soil layer | proportion |
| **ROOTR** | effective fraction of roots in each soil layer | proportion |
| **ROOTR_COLUMN** | effective fraction of roots in each soil layer | proportion |
| **RR** | root respiration (fine root MR + total root GR) | gC/m^2/s |
| **RRESIS** | root resistance in each soil layer | proportion |
| **RSSHA** | shaded leaf stomatal resistance | s/m |
| **RSSUN** | sunlit leaf stomatal resistance | s/m |
| **Rainf** | atmospheric rain | mm/s |
| **Rnet** | net radiation | W/m^2 |
| **SABG** | solar rad absorbed by ground | W/m^2 |
| **SABV** | solar rad absorbed by veg | W/m^2 |
| **SEEDC** | pool for seeding new PFTs | gC/m^2 |
| **SEEDN** | pool for seeding new PFTs | gN/m^2 |
| **SHA_AID** | shade canopy absorbed indirect from direct | W/m^2 |
| **SHA_AII** | shade canopy absorbed indirect from indirect | W/m^2 |
| **SHA_ALF** | shade canopy total absored by leaves | W/m^2 |
| **SHA_APERLAI** | shade canopy total absorbed per unit LAI | W/m^2 |
| **SHA_ATOT** | shade canopy total absorbed | W/m^2 |
| **SHA_FAID** | fraction shade canopy absorbed indirect from direct | proportion |
| **SHA_FAII** | fraction shade canopy absorbed indirect from indirect | proportion |
| **SLASHA** | specific leaf area for shaded canopy, projected area basis | m^2/gC |
| **SLASUN** | specific leaf area for sunlit canopy, projected area basis | m^2/gC |
| **SMINN** | soil mineral N | gN/m^2 |
| **SMINN_LEACHED** | soil mineral N pool loss to leaching | gN/m^2/s |
| **SMINN_TO_DENIT_EXCESS** | denitrification from excess mineral N pool | gN/m^2/s |

| Name | Long-name | Units |
|---|---|---|
| **SMINN_TO_DENIT_L1S1** | denitrification for decomp. of litter 1 to SOM 1 | gN/m^2/s |
| **SMINN_TO_DENIT_L2S2** | denitrification for decomp. of litter 2 to SOM 2 | gN/m^2/s |
| **SMINN_TO_DENIT_L3S3** | denitrification for decomp. of litter 3 to SOM 3 | gN/m^2/s |
| **SMINN_TO_DENIT_S1S2** | denitrification for decomp. of SOM 1 to SOM 2 | gN/m^2/s |
| **SMINN_TO_DENIT_S2S3** | denitrification for decomp. of SOM 2 to SOM 3 | gN/m^2/s |
| **SMINN_TO_DENIT_S3S4** | denitrification for decomp. of SOM 3 to SOM 4 | gN/m^2/s |
| **SMINN_TO_DENIT_S4** | denitrification for decomp. of SOM 4 | gN/m^2/s |
| **SMINN_TO_NPOOL** | deployment of soil mineral N uptake | gN/m^2/s |
| **SMINN_TO_PLANT** | plant uptake of soil mineral N | gN/m^2/s |
| **SMINN_TO_SOIL1N_L1** | mineral N flux for decomp. of litter 1 to SOM 1 | gN/m^2/s |
| **SMINN_TO_SOIL2N_L2** | mineral N flux for decomp. of litter 2 to SOM 2 | gN/m^2/s |
| **SMINN_TO_SOIL2N_S1** | mineral N flux for decomp. of SOM 1 to SOM 2 | gN/m^2/s |
| **SMINN_TO_SOIL3N_L3** | mineral N flux for decomp. of litter 3 to SOM 3 | gN/m^2/s |
| **SMINN_TO_SOIL3N_S2** | mineral N flux for decomp. of SOM 2 to SOM 3 | gN/m^2/s |
| **SMINN_TO_SOIL4N_S3** | mineral N flux for decomp. of SOM 3 to SOM 4 | gN/m^2/s |
| **SMP** | soil matric potential (vegetated landunits only) | mm |
| **SNOAERFRC2L** | surface forcing of all aerosols in snow, averaged only when snow is present (land) | W/m^2 |
| **SNOAERFRCL** | surface forcing of all aerosols in snow (land) | W/m^2 |
| **SNOBCFRC2L** | surface forcing of BC in snow, averaged only when snow is present (land) | W/m^2 |
| **SNOBCFRCL** | surface forcing of BC in snow (land) | W/m^2 |
| **SNOBCMCL** | mass of BC in snow column | kg/m2 |

| Name | Long-name | Units |
|---|---|---|
| **SNOBCMSL** | mass of BC in top snow layer | kg/m2 |
| **SNODSTFRC2L** | surface forcing of dust in snow, averaged only when snow is present (land) | W/m^2 |
| **SNODSTFRCL** | surface forcing of dust in snow (land) | W/m^2 |
| **SNODSTMCL** | mass of dust in snow column | kg/m2 |
| **SNODSTMSL** | mass of dust in top snow layer | kg/m2 |
| **SNOFSDSND** | direct nir incident solar radiation on snow | W/m^2 |
| **SNOFSDSNI** | diffuse nir incident solar radiation on snow | W/m^2 |
| **SNOFSDSVD** | direct vis incident solar radiation on snow | W/m^2 |
| **SNOFSDSVI** | diffuse vis incident solar radiation on snow | W/m^2 |
| **SNOFSRND** | direct nir reflected solar radiation from snow | W/m^2 |
| **SNOFSRNI** | diffuse nir reflected solar radiation from snow | W/m^2 |
| **SNOFSRVD** | direct vis reflected solar radiation from snow | W/m^2 |
| **SNOFSRVI** | diffuse vis reflected solar radiation from snow | W/m^2 |
| **SNOLIQFL** | top snow layer liquid water fraction (land) | fraction |
| **SNOOCFRC2L** | surface forcing of OC in snow, averaged only when snow is present (land) | W/m^2 |
| **SNOOCFRCL** | surface forcing of OC in snow (land) | W/m^2 |
| **SNOOCMCL** | mass of OC in snow column | kg/m2 |
| **SNOOCMSL** | mass of OC in top snow layer | kg/m2 |
| **SNORDSL** | top snow layer effective grain radius | m^-6 |
| **SNOTTOPL** | snow temperature (top layer) | K/m |
| **SNOW** | atmospheric snow | mm/s |
| **SNOWDP** | snow height | m |
| **SNOWICE** | snow ice | kg/m2 |
| **SNOWLIQ** | snow liquid water | kg/m2 |
| **SNOW_SINKS** | snow sinks (liquid water) | mm/s |

| Name | Long-name | Units |
|---|---|---|
| **SNOW_SOURCES** | snow sources (liquid water) | mm/s |
| **SNOdTdzL** | top snow layer temperature gradient (land) | K/m |
| **SOIL1C** | soil organic matter C (fast pool) | gC/m^2 |
| **SOIL1C_TO_SOIL2C** | decomp. of SOM 1 C to SOM 2 C | gC/m^2/s |
| **SOIL1N** | soil organic matter N (fast pool) | gN/m^2 |
| **SOIL1N_TO_SOIL2n** | decomp. of SOM 1 N to SOM 2 N | gN/m^2/s |
| **SOIL1_HR** | het. resp. from SOM 1 C | gC/m^2/s |
| **SOIL2C** | soil organic matter C (medium pool) | gC/m^2 |
| **SOIL2C_TO_SOIL3C** | decomp. of SOM 2 C to SOM 3 C | gC/m^2/s |
| **SOIL2N** | soil organic matter N (medium pool) | gN/m^2 |
| **SOIL2N_TO_SOIL3N** | decomp. of SOM 2 N to SOM 3 N | gN/m^2/s |
| **SOIL2_HR** | het. resp. from SOM 2 C | gC/m^2/s |
| **SOIL3C** | soil organic matter C (slow pool) | gC/m^2 |
| **SOIL3C_TO_SOIL4C** | decomp. of SOM 3 C to SOM 4 C | gC/m^2/s |
| **SOIL3N** | soil orgainc matter N (slow pool) | gN/m^2 |
| **SOIL3N_TO_SOIL4N** | decomp. of SOM 3 N to SOM 4 N | gN/m^2/s |
| **SOIL3_HR** | het. resp. from SOM 3 C | gC/m^2/s |
| **SOIL4C** | soil organic matter C (slowest pool) | gC/m^2 |
| **SOIL4N** | soil orgainc matter N (slowest pool) | gN/m^2 |
| **SOIL4N_TO_SMINN** | N mineralization for decomp. of SOM 4 | gN/m^2/s |
| **SOIL4_HR** | het. resp. from SOM 4 C | gC/m^2/s |
| **SOILC** | soil C | gC/m^2 |
| **SOILC_HR** | soil C heterotrophic respiration | gC/m^2/s |
| **SOILC_LOSS** | soil C loss | gC/m^2/s |
| **SOILICE** | soil ice (vegetated landunits only) | kg/m2 |
| **SOILLIQ** | soil liquid water (vegetated landunits only) | kg/m2 |

| Name | Long-name | Units |
|---|---|---|
| **SOILPSI** | soil water potential in each soil layer | MPa |
| **SOILWATER_10CM** | soil liquid water + ice in top 10cm of soil (veg landunits only) | kg/m2 |
| **SOMFIRE** | soil organic matter fire losses | gC/m^2/s |
| **SOMHR** | soil organic matter heterotrophic respiration | gC/m^2/s |
| **SR** | total soil respiration (HR + root resp) | gC/m^2/s |
| **STORAGE_GR** | growth resp for growth sent to storage for later display | gC/m^2/s |
| **STORVEGC** | stored vegetation carbon, excluding cpool | gC/m^2 |
| **STORVEGN** | stored vegetation nitrogen | gN/m^2 |
| **SUN_ADD** | sun canopy absorbed direct from direct | W/m^2 |
| **SUN_AID** | sun canopy absorbed indirect from direct | W/m^2 |
| **SUN_AII** | sun canopy absorbed indirect from indirect | W/m^2 |
| **SUN_ALF** | sun canopy total absorbed by leaves | W/m^2 |
| **SUN_APERLAI** | sun canopy total absorbed per unit LAI | W/m^2 |
| **SUN_ATOT** | sun canopy total absorbed | W/m^2 |
| **SUN_FAID** | fraction sun canopy absorbed indirect from direct | proportion |
| **SUN_FAII** | fraction sun canopy absorbed indirect from indirect | proportion |
| **SUPPLEMENT_TO_SMINN** | supplemental N supply | gN/m^2/s |
| **SWdown** | atmospheric incident solar radiation | W/m^2 |
| **SWup** | upwelling shortwave radiation | W/m^2 |
| **SoilAlpha** | factor limiting ground evap | unitless |
| **SoilAlpha_U** | urban factor limiting ground evap | unitless |
| **T10** | 10-day running mean of 2-m temperature | K |
| **TAUX** | zonal surface stress | kg/m/s^2 |
| **TAUY** | meridional surface stress | kg/m/s^2 |
| **TBOT** | atmospheric air temperature | K |

| Name | Long-name | Units |
|---|---|---|
| **TBUILD** | internal urban building temperature | K |
| **TDA** | daily average 2-m temperature | K |
| **TEMPAVG_T2M** | temporary average 2m air temperature | K |
| **TEMPMAX_RETRANSN** | temporary annual max of retranslocated N pool | gN/m^2 |
| **TEMPSUM_POTENTIAL_GPP** | temporary annual sum of potential GPP | gC/m^2/yr |
| **TG** | ground temperature | K |
| **TG_R** | Rural ground temperature | K |
| **TG_U** | Urban ground temperature | K |
| **THBOT** | atmospheric air potential temperature | K |
| **TLAI** | total projected leaf area index | none |
| **TLAKE** | lake temperature | K |
| **TOPT** | topt coefficient for VOC calc | non |
| **TOTCOLC** | total column carbon, incl veg and cpool | gC/m^2 |
| **TOTCOLN** | total column-level N | gN/m^2 |
| **TOTECOSYSC** | total ecosystem carbon, incl veg but excl cpool | gC/m^2 |
| **TOTECOSYSN** | total ecosystem N | gN/m^2 |
| **TOTFIRE** | total ecosystem fire losses | gC/m^2/s |
| **TOTLITC** | total litter carbon | gC/m^2 |
| **TOTLITN** | total litter N | gN/m^2 |
| **TOTPFTC** | total pft-level carbon, including cpool | gC/m^2 |
| **TOTPFTN** | total PFT-level nitrogen | gN/m^2 |
| **TOTPRODC** | total wood product C | gC/m^2 |
| **TOTPRODN** | total wood product N | gN/m^2 |
| **TOTSOMC** | total soil organic matter carbon | gC/m^2 |
| **TOTSOMN** | total soil organic matter N | gN/m^2 |
| **TOTVEGC** | total vegetation carbon, excluding cpool | gC/m^2 |
| **TOTVEGN** | total vegetation nitrogen | gN/m^2 |
| **TOT_AID** | total canopy absorbed indirect from direct | W/m^2 |
| **TRAFFICFLUX** | sensible heat flux from urban traffic | W/m^2 |

| Name | Long-name | Units |
|---|---|---|
| **TRANSFER_DEADCROOT_GR** | dead coarse root growth respiration from storage | gC/m^2/s |
| **TRANSFER_DEADSTEM_GR** | dead stem growth respiration from storage | gC/m^2/s |
| **TRANSFER_FROOT_GR** | fine root growth respiration from storage | gC/m^2/s |
| **TRANSFER_GR** | growth resp for transfer growth displayed in this timestep | gC/m^2/s |
| **TRANSFER_LEAF_GR** | leaf growth respiration from storage | gC/m^2/s |
| **TRANSFER_LIVECROOT_GR** | live coarse root growth respiration from storage | gC/m^2/s |
| **TRANSFER_LIVESTEM_GR** | live stem growth respiration from storage | gC/m^2/s |
| **TREFMNAV** | daily minimum of average 2-m temperature | K |
| **TREFMNAV_R** | Rural daily minimum of average 2-m temperature | K |
| **TREFMNAV_U** | Urban daily minimum of average 2-m temperature | K |
| **TREFMXAV** | daily maximum of average 2-m temperature | K |
| **TREFMXAV_R** | Rural daily maximum of average 2-m temperature | K |
| **TREFMXAV_U** | Urban daily maximum of average 2-m temperature | K |
| **TSA** | 2m air temperature | K |
| **TSAI** | total projected stem area index | none |
| **TSA_R** | Rural 2m air temperature | K |
| **TSA_U** | Urban 2m air temperature | K |
| **TSOI** | soil temperature (vegetated landunits only) | K |
| **TSOI_10CM** | soil temperature in top 10cm of soil | K |
| **TSOI_ICE** | soil temperature (ice landunits only) | K |
| **TV** | vegetation temperature | K |
| **TV24** | vegetation temperature (last 24hrs) | K |
| **TV240** | vegetation temperature (last 240hrs) | K |

| Name | Long-name | Units |
|---|---|---|
| **Tair** | atmospheric air temperature | K |
| **U10** | 10-m wind | m/s |
| **U10_DUST** | 10-m wind for dust model | m/s |
| **ULRAD** | upward longwave radiation above the canopy | W/m^2 |
| **URBAN_AC** | urban air conditioning flux | W/m^2 |
| **URBAN_HEAT** | urban heating flux | W/m^2 |
| **VA** | atmospheric wind speed plus convective velocity | m/s |
| **VCMXSHA** | shaded leaf Vcmax | umolCO2/m^2/s |
| **VCMXSUN** | sunlit leaf Vcmax | umolCO2/m^2/s |
| **VEGFIRE** | pft-level fire loss | gC/m^2/s |
| **VOCFLXT** | total VOC flux into atmosphere | moles/m2/sec |
| **WA** | water in the unconfined aquifer (vegetated landunits only) | mm |
| **WASTEHEAT** | sensible heat flux from heating/cooling sources of urban waste heat | W/m^2 |
| **WF** | soil water as frac. of whc for top 0.5 m | proportion |
| **WIND** | atmospheric wind velocity magnitude | m/s |
| **WOODC** | wood C | gC/m^2 |
| **WOODC_ALLOC** | wood C allocation | gC/m^2/s |
| **WOODC_LOSS** | wood C loss | gC/m^2/s |
| **WOOD_HARVESTC** | wood harvest carbon (to product pools) | gC/m^2/s |
| **WOOD_HARVESTN** | wood harvest N (to product pools) | gN/m^2/s |
| **WT** | total water storage (unsaturated soil water + groundwater, veg landunits) | mm |
| **Wind** | atmospheric wind velocity magnitude | m/s |
| **XSMRPOOL** | temporary photosynthate C pool | gC/m^2 |
| **XSMRPOOL_C13RATIO** | C13/C(12+13) ratio for xsmrpool | proportion |
| **XSMRPOOL_RECOVER** | C flux assigned to recovery of negative xsmrpool | gC/m^2/s |
| **Z0HG** | roughness length over ground, sensible heat | m |

| Name | Long-name | Units |
|---|---|---|
| **Z0HV** | roughness length over vegetation, sensible heat | m |
| **Z0M** | momentum roughness length | m |
| **Z0MG** | roughness length over ground, momentum | m |
| **Z0MV** | roughness length over vegetation, momentum | m |
| **Z0QG** | roughness length over ground, latent heat | m |
| **Z0QV** | roughness length over vegetation, latent heat | m |
| **ZBOT** | atmospheric reference height | m |
| **ZII** | convective boundary height | m |
| **ZWT** | water table depth (vegetated landunits only) | m |
| **aais_area** | Antarctic ice area | km^2 |
| **aais_mask** | Antarctic mask | unitless |
| **gris_area** | Greenland ice area | km^2 |
| **gris_mask** | Greenland mask | unitless |

# Examples of using different namelist features

Below we will give examples of user namelists that activate different commonly used namelist features. We will discuss the namelist features in different examples and then show a user namelist that includes an example of the use of these features. First we will show the default namelist that doesn't activate any user options.

## The default namelist

Here we give the default namelist as it would be created for a I1850CN compset at 0.9x1.25 resolution with a gx1v6 land-mask. To edit the namelist you would edit the `user_nl_clm` user namelist with just the items you want to change. For simplicity we will just show the CLM namelist and NOT the entire file. In the sections below, for simplicity we will just show the user namelist (`user_nl_clm`) that will add (or modify existing) namelist items to the namelist.

**Example 1-2. Default CLM Namelist**

```
&clm_inparm
 co2_ppmv   = 284.7
 co2_type   = 'constant'
 create_crop_landunit = .false.
 dtime   = 1800
 fatmlndfrc = '$DIN_LOC_ROOT/share/domains/domain.lnd.fv0.9x1.25_gx1v6.090309.nc'
 finidat = '$DIN_LOC_ROOT/ccsm4_init/b40.1850.track1.1deg.006/0863-01-01/b40.1850.track1.1deg.006.clm2.r.0863-01-01-00000.nc'
```

```
 fpftcon    = '$DIN_LOC_ROOT/lnd/clm2/pftdata/pft-physiology.c110425.nc'
 fsnowaging   =
'$DIN_LOC_ROOT/lnd/clm2/snicardata/snicar_drdt_bst_fit_60_c070416.nc'
 fsnowoptics =
'$DIN_LOC_ROOT/lnd/clm2/snicardata/snicar_optics_5bnd_c090915.nc'
 fsurdat    =
'$DIN_LOC_ROOT/lnd/clm2/surfdata/surfdata_0.9x1.25_simyr1850_c110921.nc'
 maxpatch_glcmec = 0
 nsegspc = 20
 urban_hac = 'ON_WASTEHEAT'
 urban_traffic = .false.
/
&ndepdyn_nml
 ndepmapalgo = 'bilinear'
 stream_fldfilename_ndep    =
'$DIN_LOC_ROOT/lnd/clm2/ndepdata/fndep_clm_hist_simyr1849-2006_1.9x2.5_c100428.nc'
 stream_year_first_ndep   = 1850
 stream_year_last_ndep   = 1850
/
```

## Adding/removing fields on your primary history file

The primary history files are output monthly, and contain an extensive list of fieldnames, but the list of fieldnames can be added to using `hist_fincl1` or removed from by adding fieldnames to `hist_fexcl1`. A sample user namelist `user_nl_clm` adding few new fields (cosine of solar zenith angle, and solar declination) and excluding a few standard fields is (ground temperature, vegetation temperature, soil temperature and soil water).:

**Example 1-3. Example `user_nl_clm` namelist adding and removing fields on primary history file**

```
hist_fincl1 = 'COSZEN', 'DECL'
hist_fexcl1 = 'TG', 'TV', 'TSOI', 'H2OSOI'
```

## Adding auxiliary history files and changing output frequency

The `hist_fincl2` through `hist_fincl6` set of namelist variables add given history fieldnames to auxiliary history file "streams", and `hist_fexcl2` through `hist_fexcl6` set of namelist variables remove given history fieldnames from history file auxiliary "streams". A history "stream" is a set of history files that are produced at a given frequency. By default there is only one stream of monthly data files. To add more streams you add history fieldnames to `hist_fincl2` through `hist_fincl6`. The output frequency and the way averaging is done can be different for each history file stream. By default the primary history files are monthly and any others are daily. You can have up to six active history streams, but you need to activate them in order. So if you activate stream "6" by setting `hist_fincl6`, but if any of `hist_fincl2` through `hist_fincl5` are unset, only the history streams up to the first blank one will be activated.

The frequency of the history file streams is given by the namelist variable `hist_nhtfrq` which is an array of rank six for each history stream. The values of the array `hist_nhtfrq` must be integers, where the following values have the given meaning:

*Positive value* means the output frequency is the number of model steps between output.
*Negative value* means the output frequency is the absolute value in hours given (i.e -1 would mean an hour and -24 would
*Zero* means the output frequency is monthly. This is the default for the primary history files.

The number of samples on each history file stream is given by the namelist variable `hist_mfilt` which is an array of rank six for each history stream. The values of the array `hist_mfilt` must be positive integers. By default the primary history file stream has one time sample on it (i.e. output is to separate monthly files), and all other streams have thirty time samples on them.

A sample user namelist `user_nl_clm` turning on four extra file streams for output: daily, six-hourly, hourly, and every time-step, leaving the primary history files as monthly, and changing the number of samples on the streams to: yearly (12), thirty, weekly (28), daily (24), and daily (48) is:

**Example 1-4. Example `user_nl_clm` namelist adding auxiliary history files and changing output frequency**

```
hist_fincl2 = 'TG', 'TV'
hist_fincl3 = 'TG', 'TV'
hist_fincl4 = 'TG', 'TV'
hist_fincl5 = 'TG', 'TV'
hist_nhtfrq = 0, -24, -6, -1, 1
hist_mfilt  = 12, 30, 28, 24, 48
```

## Removing all history fields

Sometimes for various reasons you want to remove all the history fields either because you want to do testing without any output, or you only want a very small custom list of output fields rather than the default extensive list of fields. By default only the primary history files are active, so technically using `hist_fexcl1` explained in the first example, you could list *ALL* of the history fields that are output in `hist_fexcl1` and then you wouldn't get any output. However, as the list is very extensive this would be a cumbersome thing to do. So to facilitate this `hist_empty_htapes` allows you to turn off all default output. You can still use `hist_fincl1` to turn your own list of fields on, but you then start from a clean slate. A sample user namelist `user_nl_clm` turning off all history fields and then activating just a few selected fields (ground and vegetation temperatures and absorbed solar radiation) is:

**Example 1-5. Example `user_nl_clm` namelist removing all history fields**

```
hist_empty_htapes = .true.
hist_fincl1 = 'TG', 'TV', 'FSA'
```

Note, you could also build adding the "-noio" option to CLM_CONFIG_OPTS. But, this would build the model without history output and you wouldn't be able to add that in later.

## Various ways to change history output averaging flags

There are two ways to change the averaging of output history fields. The first is using `hist_avgflag_pertape` which gives a default value for each history stream, the second is when you add fields using `hist_fincl*`, you add an averaging flag to the end of the field name after a colon (for example 'TSOI:X', would output the maximum of TSOI). The types of averaging that can be done are:

*A* Average, over the output interval.
*I* Instantaneous, output the value at the output interval.
*X* Maximum, over the output interval.
*M* Minimum, over the output interval.

The default averaging depends on the specific fields, but for most fields is an average. A sample user namelist `user_nl_clm` making the monthly output fields all averages (except TSOI for the first two streams and FIRE for the 5th stream), and adding auxiliary file streams for instantaneous (6-hourly), maximum (daily), minimum (daily), and average (daily). For some of the fields we diverge from the per-tape value given and customize to some different type of optimization.

**Example 1-6. Example `user_nl_clm` namelist with various ways to average history fields**

```
hist_empty_htapes = .true.
hist_fincl1 = 'TSOI:X', 'TG',   'TV',   'FIRE',   'FSR', 'FSH',
              'EFLX_LH_TOT', 'WT'
hist_fincl2 = 'TSOI:X', 'TG',   'TV',   'FIRE',   'FSR', 'FSH',
              'EFLX_LH_TOT', 'WT'
hist_fincl3 = 'TSOI',   'TG:I', 'TV',   'FIRE',   'FSR', 'FSH',
              'EFLX_LH_TOT', 'WT'
hist_fincl4 = 'TSOI',   'TG',   'TV:I', 'FIRE',   'FSR', 'FSH',
              'EFLX_LH_TOT', 'WT'
hist_fincl5 = 'TSOI',   'TG',   'TV',   'FIRE:I', 'FSR', 'FSH',
              'EFLX_LH_TOT', 'WT'
hist_avgflag_pertape = 'A', 'I', 'X',   'M', 'A'
hist_nhtfrq = 0, -6, -24, -24, -24
```

In the example we put the same list of fields on each of the tapes: soil-temperature, ground temperature, vegetation temperature, emitted longwave radiation, reflected solar radiation, sensible heat, total latent-heat, and total water storage. We also modify the soil-temperature for the primary and secondary auxiliary tapes by outputting them for a maximum instead of the prescribed per-tape of average and instantaneous respectively. For the tertiary auxiliary tape we output ground temperature instantaneous instead of as a maximum, and for the fourth auxiliary tape we output vegetation temperature instantaneous instead of as a minimum. Finally, for the fifth auxiliary tapes we output `FIRE` instantaneously instead of as an average.

> **Note:** We also use `hist_empty_htapes` as in the previous example, so we can list ONLY the fields that we want on the primary history tapes.

## Outputting history files as a vector in order to analyze the plant function types within gridcells

By default the output to history files are the grid-cell average of all land-units, and vegetation types within that grid-cell, and output is on the full 2D latitude/longitude grid with ocean masked out.

Sometimes it's important to understand how different land-units or vegetation types are acting within a grid-cell. The way to do this is to output history files as a 1D-vector of all land-units and vegetation types. In order to display this, you'll need to do extensive post-processing to make sense of the output. Often you may only be interested in a few points, so once you figure out the 1D indices for the grid-cells of interest, you can easily view that data. 1D vector output can also be useful for single point datasets, since it's then obvious that all data is for the same grid cell.

To do this you use `hist_dov2xy` which is an array of rank six for each history stream. Set it to `.false.` if you want one of the history streams to be a 1D vector. You can also use `hist_type1d_pertape` if you want to average over all the: Plant-Function-Types, columns, land-units, or grid-cells. A sample user namelist `user_nl_clm` leaving the primary monthly files as 2D, and then doing grid-cell (GRID), column (COLS), and no averaging over auxiliary tapes output daily for a single field (ground temperature) is:

**Example 1-7. Example `user_nl_clm` namelist outputting some files in 1D Vector format**

```
hist_fincl2 = 'TG'
hist_fincl3 = 'TG'
hist_fincl4 = 'TG'
hist_fincl5 = 'TG'
hist_fincl6 = 'TG'
hist_dov2xy = .true., .false., .false., .false.
hist_type2d_pertape = ' ', 'GRID', 'COLS', ' '
hist_nhtfrq = 0, -24, -24, -24
```

## Warning

LAND and COLS are also options to the pertape averaging, but currently there is a bug with them and they fail to work.

**Note:** Technically the default for `hist_nhtfrq` is for primary files output monthly and the other auxiliary tapes for daily, so we don't actually have to include `hist_nhtfrq`, we could use the default for it. Here we specify it for clarity.

## Caution

Visualizing global 1D vector files will take effort. You'll probably want to do some post-processing and possibly just extract out single points of interest to see what is going on. Since, the output is a 1D vector, of only land-points traditional plots won't be helpful. The number of points per grid-cell will also vary for anything, but grid-cell averaging. You'll need to use the output fields `pfts1d_ixy`, and `pfts1d_jxy`, to get the mapping of the fields to the global 2D array. `pfts1d_itype_veg` gives you the PFT number for each PFT. Most likely you'll want to do this analysis in a data processing tool (such as NCL, Matlab, Mathmatica, IDL, etcetera that is able to read and process NetCDF data files).

### Conclusion to namelist examples

We've given various examples of namelists that feature the use of different namelist options to customize a case for particular uses. Most the examples revolve around how to customize the output history fields. This should give you a good basis for setting up your own CLM namelist.

# Customizing the DATM Namelist and Streams files

When running "I" compsets with CLM you use the DATM model to give atmospheric forcing data to CLM. There are four ways to customize DATM:

1. *DATM Main Namelist* (`datm_in`)
2. *DATM Stream Namelist* (`datm_atm_in`)
3. *DATM stream files*
4. *DATM build namelist file* (`Buildconf/`**`datm.buildnml.csh`**)

The Data Model Documentation (http://www.cesm.ucar.edu/models/cesm1.1//data8/doc/book1.html) gives the details of all the options for the data models and for DATM specifically. It goes into detail on all namelist items both for DATM and for DATM streams. So here we won't list ALL of the DATM namelist options, nor go into great details about stream files. But, we will talk about a few of the different options that are relevant for running with CLM. All of the options for changing the namelists or stream files is done by editing the `user_nl_datm` file.

Because, they aren't useful for work with CLM we will NOT discuss any of the options for the main DATM namelist. Use the DATM Users Guide at the link above to find details of that. For the streams namelist we will discuss three items:

1. *mapalgo*
2. *taxmode*
3. *tintalgo*

And for the streams file itself we will discuss:

*offset*

Again everything else (and including the above items) are discussed in the Data Model User's Guide. Of the above the last three: offset, taxmode and tintalgo are all closely related and have to do with the time interpolation of the DATM data.

mapalgo

`mapalgo` sets the spatial interpolation method to go from the DATM input data to the output DATM model grid. The default is `bilinear`. For `CLM1PT` we set it to `nn` to just select the nearest neighbor. This saves time and we also had problems running the interpolation for single-point mode.

taxmode

`taxmode` is the time axis mode. For CLM we usually have it set to `cycle` which means that once the end of the data is reached it will start over at the beginning. The `extend` modes is used have it use the last time-step of the forcing data once it reaches the end of forcing data (or use the first time-step before it reaches where the forcing data starts). See the warning below about the `extend` mode.

**Warning**

*THE `extend` OPTION NEEDS TO BE USED WITH CAUTION!* It is only invoked by default for the `CLM1PT` mode and is only intended for the supported urban datasets to extend the data for a single time-step. If you have the model *run extensively through periods in this mode you will effectively be repeating that last time-step over that entire period*. This means the output of your simulation will be worthless.

offset (in the stream file)

`offset` is the time offset in seconds to give to each stream of data. Normally it is NOT used because the time-stamps for data is set correctly for each stream of data. Note, the `offset` may NEED to be adjusted depending on the `taxmode` described above, or it may need to be adjusted to account for data that is time-stamped at the END of an interval rather than the middle or beginning of interval. The `offset` can is set in the stream file rather than on the stream namelist. For data with a `taxmode` method of `coszen` the time-stamp needs to be for the beginning of the interval, while for other data it should be the midpoint. The `offset` can be used to adjust the time-stamps to get the data to line up correctly.

tintalgo

`tintalgo` is the time interpolation algorithm. For CLM we usually use one of three modes: `coszen`, `nearest`, or `linear`. We use `coszen` for solar data, `nearest` for precipitation data, and `linear` for everything else. If your data is half-hourly or hourly, `nearest` will work fine for everything. The `coszen` scaling is useful for longer periods (three hours or more) to try to get the solar to match the cosine of the solar zenith angle over that longer period of time. If you use `linear` for longer intervals, the solar will cut out at night-time anyway, and the straight line will be a poor approximation of the cosine of the solar zenith angle of actual solar data. `nearest` likewise would be bad for longer periods where it would be much higher than the actual values.

**Note:** For `coszen` the time-stamps of the data should correspond to the beginning of the interval the data is measured for. Either make sure the time-stamps on the datafiles is set this way, or use the `offset` described above to set it.

**Note:** For `nearest` and `linear` the time-stamps of the data should correspond to the middle of the interval the data is measured for. Either make sure the time-stamps on the datafiles is set this way, or use the `offset` described above to set it.

In the sections below we go over each of the relevant DATM_MODE options and what the above DATM settings are for each. This gives you examples of actual usage for the settings. We also describe in what ways you might want to customize them for your own case.

## CLM_QIAN mode and it's DATM settings

In CLM_QIAN mode the Qian dataset is used which has 6-hourly solar and precipitation data, and 3-hourly for everything else. The dataset is divided into those three data streams: solar, precipitation, and everything else (temperature, pressure, humidity and wind). The time-stamps of the data were also adjusted so that they are the beginning of the interval for solar, and the middle for the other two. Because, of this the offset is set to zero, and the tintalgo is: coszen, nearest, and linear for the solar, precipitation and other data respectively. taxmode is set to cycle and mapalgo is set to bilinear so that the data is spatially interpolated from the input T62 grid to the grid the atmosphere model is being run at.

Normally you wouldn't customize the CLM_QIAN settings, but you might replicate it's use for your own global data that had similar temporal characteristics.

## CLM1PT mode and it's DATM settings

In CLM1PT mode the model is assumed to have half-hourly or hourly data for a single-point. For the supported datasets that is exactly what it has. But, if you add your own data you may need to make adjustments accordingly. Using the CLM_USRDAT_NAME resolution you can easily extend this mode for your own datasets that may be regional or even global and could be at different temporal frequencies. If you do so you'll need to make adjustments to your DATM settings. The dataset has all data in a single stream file. The time-stamps of the data were also adjusted so that they are at the middle of the interval. Because, of this the offset is set to zero, and the tintalgo is set to nearest. taxmode is set to extend and mapalgo is set to nn so that simply the nearest point is used.

If you are using your own data for this mode and it's not at least hourly you'll want to adjust the DATM settings for it. If the data is three or six hourly, you'll need to divide it up into separate streams like in CLM_QIAN mode which will require fairly extensive changes to the DATM namelist and streams files. For an example of doing this see Example 5-7.

## CPLHIST3HrWx mode and it's DATM settings

In CPLHIST3HrWx mode the model is assumed to have 3-hourly for a global grid from a previous CESM simulation. Like CLM_QIAN mode the data is divided into three streams: one for precipitation, one for solar, and one for everything else. The time-stamps for Coupler history files for CESM is at the end of the interval, so the offset needs to be set in order to adjust the time-stamps to what it needs to be for the tintalgo settings. For precipitation taxmode is set to nearest so the offset is set to -5400 seconds so that the ending time-step is adjusted by an hour and half to the middle of the interval. For solar taxmode is set to coszen so the offset is set to -10800 seconds so that the ending time-step is adjust by three hours to the beginning of the interval. For everything else taxmode is set to linear so the offset is set to -5400 seconds so that the ending time-step is adjusted by an hour and half to the middle of the interval.

Normally you wouldn't modify the DATM settings for this mode. However, if you had data at a different frequency than 3-hours you would need to modify the `offset` and possibly the `taxmode`. The other two things that you might modify would be the path to the data or the domain file for the resolution (which is currently hardwired to f09). For data at a different input resolution you would need to change the domain file in the streams file to use a domain file to the resolution that the data comes in on.

# Conclusion to customizing chapter

We've given extensive details on customizing cases with CLM, by choosing compsets, by changing env options and interacting with the CLM "**configure**" and "**build-namelist**" scripts, we've given details on all of the CLM namelist items, and finally given some instruction in customizing the DATM namelist and streams files. In the next chapter we talk about further ways to customize cases with CLM by creating your own datasets using the tools provided in CLM.

# Chapter 2. Using the CLM tools to create your own input datasets

There are several tools provided with CLM that allow you to create your own input datasets at resolutions you choose, or to interpolate initial conditions to a different resolution, or used to compare CLM history files between different cases. The tools are all available in the `models/lnd/clm/tools` directory. Most of the tools are FORTRAN stand-alone programs in their own directory, but there is also a suite of NCL scripts in the `ncl_scripts` directory, and some of the tools are scripts that may also call the ESMF regridding program. Some of the NCL scripts are very specialized and not meant for general use, and we won't document them here. They still contain documentation in the script itself and the README file in the tools directory. But, the list of generally important scripts and programs are:

1. *cprnc* to compare NetCDF files with a time axis.
2. *mkmapgrids* to create SCRIP grid data files from old CLM format grid files that can then be used to create new CLM datasets.

   *mkmapdata* to create SCRIP mapping data file from SCRIP grid files (uses ESMF).
3. *gen_domain* to create a domain file for datm from a mapping file. The domain file is then used by BOTH datm AND CLM to define the grid and land-mask.
4. *mksurfdata_map* to create surface datasets from grid datasets.
5. *interpinic* to interpolate initial condition files.
6. *ncl_scripts/getregional_datasets.pl* script to extract a region or a single-point from global input datasets. See the single-point chapter for more information on this.
7. *mkprocdata_map* to interpolate output unstructured grids (such as the CAM HOMME dy-core "ne" grids like ne30np4) into a 2D regular lat/long grid format that can be plotted easily.

In the sections to come we will go into detailed description of how to use each of these tools in turn. First, however we will discuss the common environment variables and options that are used by all of the FORTRAN tools. Second, we go over the outline of the entire file creation process for all input files needed by CLM for a new resolution, then we turn to each tool. In the last section we will discuss how to customize files for particular observational sites.

## Common environment variables and options used in building the FORTRAN tools

The FORTRAN tools all have similar makefiles, and similar options for building. All of the Makefiles use GNU Make extensions and thus require that you use GNU make to use them. They also auto detect the type of platform you are on, using "uname -s" and set the compiler, compiler flags and such accordingly. There are also environment variables that can be set to set things that must be customized. All the tools use NetCDF and hence require the path to the NetCDF libraries and include files. On some

platforms (such as Linux) multiple compilers can be used, and hence there are env variables that can be set to change the FORTRAN and/or "C" compilers used. The tools other than **cprnc** also allow finer control, by also allowing the user to add compiler flags they choose, for both FORTRAN and "C", as well as picking the compiler, linker and and add linker options. Finally the tools other than **cprnc** allow you to turn optimization on (which is off by default but on for the **mksurfdata_map** and **interpinic** programs) with the OPT flag so that the tool will run faster. To get even faster performance, the **interpinic**, program allows you to also use the SMP to turn on multiple shared memory processors. When SMP=TRUE you set the number of threads used by the program with the OMP_NUM_THREADS environment variable.

Options used by all: **cprnc**, **interpinic**, and **mksurfdata_map**

LIB_NETCDF -- sets the location of the NetCDF library.
INC_NETCDF -- sets the location of the NetCDF include files.
USER_FC -- sets the name of the FORTRAN compiler.

Options used by: **interpinic**, **mkprocdata_map**, **mkmapgrids**, and **mksurfdata_map**
MOD_NETCDF -- sets the location of the NetCDF FORTRAN module.
USER_LINKER -- sets the name of the linker to use.
USER_CPPDEFS -- adds any CPP defines to use.
USER_CFLAGS -- add any "C" compiler flags to use.
USER_FFLAGS -- add any FORTRAN compiler flags to use.
USER_LDFLAGS -- add any linker flags to use.
USER_CC -- sets the name of the "C" compiler to use.
OPT -- set to TRUE to compile the code optimized (TRUE or FALSE)
SMP -- set to TRUE to turn on shared memory parallelism (i.e. OpenMP) (TRUE or FALSE)
`Filepath` -- list of directories to build source code from.
`Srcfiles` -- list of source code filenames to build executable from.
`Makefile` -- customized makefile options for this particular tool.
`Makefile.common` -- General tool Makefile that should be the same between all tools.
Options used only by cprnc:
EXEDIR -- sets the location where the executable will be built.
VPATH -- colon delimited path list to find the source files.
More details on each environment variable.

LIB_NETCDF

This variable sets the path to the NetCDF library file (`libnetcdf.a`). If not set it defaults to `/usr/local/lib`. In order to use the tools you need to build the NetCDF library and be able to link to it. In order to build the model with a particular compiler you may have to compile the NetCDF library with the same compiler (or at least a compatible one).

INC_NETCDF

This variable sets the path to the NetCDF include directory (in order to find the include file `netcdf.inc`). if not set it defaults to `/usr/local/include`.

MOD_NETCDF

This variable sets the path to the NetCDF module directory (in order to find the NetCDF FORTRAN-90 module file when NetCDF is used with a FORTRAN-90 **use statement**. When not set it defaults to the LIB_NETCDF value.

USER_FC

This variable sets the command name to the FORTRAN-90 compiler to use when compiling the

tool. The default compiler to use depends on the platform. And for example, on the AIX platform this variable is NOT used

USER_LINKER

This variable sets the command name to the linker to use when linking the object files from the compiler together to build the executable. By default this is set to the value of the FORTRAN-90 compiler used to compile the source code.

USER_CPPDEFS

This variable adds additional optional values to define for the C preprocessor. Normally, there is no reason to do this as there are very few CPP tokens in the CLM tools. However, if you modify the tools there may be a reason to define new CPP tokens.

USER_CC

This variable sets the command name to the "C" compiler to use when compiling the tool. The default compiler to use depends on the platform. And for example, on the AIX platform this variable is NOT used

USER_CFLAGS

This variable adds additional compiler options for the "C" compiler to use when compiling the tool. By default the compiler options are picked according to the platform and compiler that will be used.

USER_FFLAGS

This variable adds additional compiler options for the FORTRAN-90 compiler to use when compiling the tool. By default the compiler options are picked according to the platform and compiler that will be used.

USER_LDFLAGS

This variable adds additional options to the linker that will be used when linking the object files into the executable. By default the linker options are picked according to the platform and compiler that is used.

SMP

This variable flags if shared memory parallelism (using OpenMP) should be used when compiling the tool. It can be set to either `TRUE` or `FALSE`, by default it is set to `FALSE`, so shared memory parallelism is NOT used. When set to `TRUE` you can set the number of threads by using the OMP_NUM_THREADS environment variable. Normally, the most you would set this to would be to the number of on-node CPU processors. Turning this on should make the tool run much faster.

**Caution**

Note, that depending on the compiler answers may be different when SMP is activated.

OPT

This variable flags if compiler optimization should be used when compiling the tool. It can be set to either TRUE or FALSE, by default it is set to FALSE for **mkmapgrids** and TRUE for **mksurfdata_map**, **mkprocdata_map** and **interpinic**. Turning this on should make the tool run much faster.

**Caution**

Note, you should expect that answers will be different when OPT is activated.

Filepath

All of the tools are stand-alone and don't need any outside code to operate. The Filepath is the list of directories needed to compile and hence is always simply "." the current directory. Several tools use copies of code outside their directory that is in the CESM distribution (either csm_share code or CLM source code).

Srcfiles

The Srcfiles lists the filenames of the source code to use when building the tool.

Makefile

The Makefile is the custom GNU Makefile for this particular tool. It will customize the EXENAME and the optimization settings for this particular tool.

Makefile.common

The Makefile.common is the copy of the general GNU Makefile for all the CLM tools. This file should be identical between the different tools. This file has different sections of compiler options for different Operating Systems and compilers.

EXEDIR

The **cprnc** tool uses this variable to set the location of where the executable will be built. The default is the current directory.

VPATH

The **cprnc** tool uses this variable to set the colon delimited pathnames of where the source code exists. The default is the current directory.

**Note:** There are several files that are copies of the original files from either models/lnd/clm/src/main, models/csm_share/shr, or copies from other tool directories. By having copies the tools can all be made stand-alone, but any changes to the originals will have to be put into the tool directories as well.

The *README.filecopies* (which can be found in models/lnd/clm/tools) is repeated here.

```
models/lnd/clm/tools/README.filecopies         Oct/13/2012

There are several files that are copies of the original files from
either models/lnd/clm/src/main, models/csm_share/shr,
models/csm_share/unit_testers, or copies from other tool
directories. By having copies the tools can all be made stand-alone,
but any changes to the originals will have to be put into the tool
directories as well.

I. Files that are IDENTICAL:

   1. csm_share files copied that should be identical to models/csm_share/shr:

       shr_kind_mod.F90
       shr_const_mod.F90
       shr_log_mod.F90
       shr_file_mod.F90

   2. csm_share files copied that should be identical to models/csm_share/unit_testers:

       test_mod.F90

II. Files with differences

   1. csm_share files copied with differences:

       shr_sys_mod.F90 - Remove mpi abort and reference to shr_mpi_mod.F90.

   2. clm/src files with differences:

       fileutils.F90 --- Remove use of masterproc and spmdMod and endrun in abortutils.

   3. Files in mkgriddata (different from mkmapgrids)

      domainMod.F90 ---- Highly customized based off an earlier version of clm code.
                         Remove use of abortutils, spmdMod. clm version uses latlon
                         this version uses domain in names. Distributed memory
                         parallelism is removed.

   4. Files in mkmapgrids (different from mkgriddata)

      domainMod.F90 ---- Highly customized based off an earlier version of clm code.
                         Remove use of abortutils, spmdMod. clm version uses latlon
                         this version uses domain in names. Distributed memory
                         parallelism is removed.

   5. Files in mksurfdata_map

       mkvarpar.F90
       clm_varctl.F90
       clm_varpar.F90
```

# General information on running the FORTRAN tools

The tools run either one of two ways, with a namelist to provide options, or with command line arguments (and NOT both). **interpinic**, **gen_domain** and **cprnc** run with command line arguments, and the other tools run with namelists.

## Running FORTRAN tools with namelists

**mksurfdata_map** and **mkmapgrids** run with namelists that are read from standard input. Hence, you create a namelist and then run them by redirecting the namelist file into standard input as follows:

```
./program < namelist
```

For programs with namelists there is at least one sample namelist with the name "program".namelist (i.e. `mksurfdata_map.namelist` for the **mksurfdata_map** program). There may also be other sample namelists that end in a different name besides "namelist". Namelists that you create should be similar to the example namelist. The namelist values are also documented along with the other namelists in the: `models/lnd/clm/bld/namelist_files/namelist_definition.xml` (../../bld/namelist_files/namelist_definition.xml) file and default values in the: `models/lnd/clm/bld/namelist_files/namelist_defaults_clm_tools.xml` (../../bld/namelist_files/namelist_defaults_clm_tools.xml) file.

## Running FORTRAN tools with command line options

**interpinic**, **gen_domain**, and **cprnc** run with command line arguments. The detailed sections below will give you more information on the command line arguments specific to each tool. Also running the tool without any arguments will give you a general synopsis on how to run the tool. For example to get help on running **interpinic** do the following.

```
cd models/lnd/clm/tools/interpinic
gmake
./interpinic
```

## Running FORTRAN tools built with SMP=TRUE

When you enable SMP=TRUE on your build of one of the tools that make use of it, you are using OpenMP for shared memory parallelism (SMP). In SMP loops are run in parallel with different threads run on different processors all of which access the same memory (called on-node). Thus you can only usefully run up to the number of processors that are available on a single-node of the machine you are running on. For example, on the NCAR machine yellowstone there are 16 processors per node, but the SMT hardware on the machine allows you to submit twice as many threads or 32 threads. So to run **interpinic** on yellowstone optimized, with 32 threads you would do the following:

```
cd models/lnd/clm/tools/interpinic/src
gmake OPT=TRUE SMP=TRUE
setenv OMP_NUM_THREADS 32
cd ..
./interpinic `cat interpinic.runoptions`
```

# Using NCL scripts

In the tools directory `models/lnd/clm/tools/ncl_scripts` and in a few other locations there are scripts that use NCAR Command Language (NCL). Unlike the FORTRAN tools, you will need to get a copy of NCL in order to use them. You also won't have to build an executable in order to use them, hence no Makefile is provided. NCL is provided for free download as either binaries or source code from: http://www.ncl.ucar.edu/. The NCL web-site also contains documentation on NCL and it's use. These scripts are stand-alone and at most use environment variables to control how to use them. In some cases there are perl scripts with command line arguments that call the NCL scripts to control what they do.

# The File Creation Process

When just creating a replacement file for an existing one, the relevant tool should be used directly to create the file. When you are creating a set of files for a new resolution there are some dependencies between the tools that you need to keep in mind when creating them. The main dependency is that you MUST create a SCRIP grid file first as the SCRIP grid dataset is then input into the other tools. Also look at Table 3-1 which gives information on the files required and when. Figure 2-1 shows an overview of the

general data-flow for creation of the `fsurdat` datasets.

**Figure 2-1. Data Flow for Creation of Surface Datasets from Raw SCRIP Grid Files**

Starting from a SCRIP grid file that describes the grid you will run the model on, you first run **mkmapdata.sh** to create a list of mapping files. The mapping files tell **mksurfdata_map** how to map between the output grid and the raw datasets that it uses as input. The output of **mksurfdata_map** is a surface dataset that you then use for running the model. See Figure 2-5 for a more detailed view of how **mksurfdata_map** works.

Figure 2-2 is the legend for this figure (Figure 2-1) and other figures in this chapter (Figure 2-3, Figure 2-4, and Figure 2-5).

**Figure 2-2. Legend for Data Flow Figures**

Green arrows define the input to a program, while red arrows define the output. Cylinders define files that are either created by a program or used as input for a program. Boxes are programs.

You start with a description of a SCRIP grid file for your output grid file and then create mapping files from the raw datasets to it. Once, the mapping files are created **mksurfdata_map** is run to create the surface dataset to run the model.

**Creating a Complete Set of Files for Input to CLM**

1. Create SCRIP grid datasets (if NOT already done)

   First you need to create a desciptor file for your grid, that includes the locations of cell centers and cell corners. There is also a "mask" field, but in this case the mask is set to one everywhere (i.e. all of the masks for the output model grid are "nomask"). An example SCRIP grid file is: `$CSMDATA/lnd/clm2/mappingdata/grids/SCRIPgrid_10x15_nomask_c110308.nc`. The **mkmapgrids** and `mkscripgrid.ncl` NCL script in the `models/lnd/clm/tools/mkmapgrids` directory can help you with this. SCRIP grid files for all the standard CLM grids are already created for you. See the Section called *Creating an output SCRIP grid file at a resolution to run the model on* for more information on this.

2. Create domain dataset (if NOT already done)

   Next use **gen_domain** to create a domain file for use by DATM and CLM. This is required, unless a domain file was already created. See the Section called *Creating a domain file for CLM and DATM* for more information on this.

3. Create mapping files for **mksurfdata_map** (if NOT already done)

   Create mapping files for **mksurfdata_map** with **mkmapdata.sh** in `models/lnd/clm/tools/mkmapdata`. See the Section called *Creating mapping files that **mksurfdata_map** will use* for more information on this.

4. Create surface datasets

   Next use **mksurfdata_map** to create a surface dataset, using the mapping datasets created on the previous step as input. See the Section called *Using mksurfdata_map to create surface datasets from grid datasets* for more information on this.

5. Create some sort of initial condition dataset

   You then need to do one of the following three options to have an initial dataset to start from.

   a. Use spinup-procedures to create initial condition datasets

      The first option is to do the spinup procedures from arbitrary initial conditions to get good initial datasets. This is the most robust method to use. See the Section called *Spinning up the Satellite Phenology Model (CLMSP spinup)* in Chapter 4, the Section called *Spinning up the biogeochemistry Carbon-Nitrogen Model (CN spinup)* in Chapter 4, or the Section called *Spinning up the Carbon-Nitrogen Dynamic Global Vegetation Model (CNDV spinup)* in Chapter 4 for more information on this.

   b. Use **interpinic** to interpolate existing initial condition datasets

The next option is to interpolate from spunup datasets at a different resolution, using **interpinic**. See the Section called *Using **interpinic** to interpolate initial conditions to different resolutions* for more information on this.

c. Start up from arbitrary initial conditions

The last alternative is to run from arbitrary initial conditions without using any spun-up datasets. This is inappropriate when using CLMCN (bgc=cn or cndv) as it takes a long time to spinup Carbon pools.

**Warning**

This is NOT recommended as many fields in CLM take a long time to equilibrate.

6. Enter the new datasets into the **build-namelist** XML database

The last optional thing to do is to enter the new datasets into the **build-namelist** XML database. See Chapter 3 for more information on doing this. This is optional because the user may enter these files into their namelists manually. The advantage of entering them into the database is so that they automatically come up when you create new cases.

The `models/lnd/clm/tools/README` (../../tools/README) goes through the complete process for creating input files needed to run CLM. We repeat that file here:

```
models/lnd/clm/tools/README     Oct/13/2012

CLM tools for analysis of CLM history files -- or for creation or
modification of CLM input files.

I.  General directory structure:

        SVN_EXTERNAL_DIRECTORIES - File to use to point to external tools (such as cprnc)
 mksurfdata_map --- Create surface datasets.
        mkmapgrids ------- Create SCRIP grid files needed by mkmapdata
                           [input is CLM grid files]
                           (deprecated)
        mkmapdata -------- Create SCRIP mapping data from SCRIP grid files (uses ESMF)
        gen_domain ------- Create data model domain datasets from SCRIP mapping datasets.
                           (also in the top level mapping directory [../../../../mapping])
 interpinic ------- Interpolate initial datasets to a different resolution.
                           (has optimized and OMP options)
 cprnc ------------ Compare output history files. (stand-alone tool)
        mkprocdata_map --- Convert output unstructured grids into a 2D format that
                           can be plotted easily
        ncl_scripts ------ NCL post or pre processing scripts.

II. Notes on building/running for each of the above tools:

    Each tool that has FORTRAN source code (except cprnc as it's stand-alone) has
    the following files:

 README ------- Specific help for using the specific tool and help on specific
                    files in that directory.
 src/Filepath ------- List of directories needed to build the tool
                           (some files in ../src directories are required).
 src/Makefile ------- GNU Makefile to build the tool
 src/Makefile.common  Common part of the Makefile between tools
                           (these are identical between tools.
 src/Srcfiles ------- List of source files that are needed.
        src/Mkdepends ------ Dependency generator program

    cprnc is a tool shared by other CESM components as it's a general tool for comparing
    NetCDF files with a time coordinate. As such it has it's own stand-alone Makefile
    and is a little different than the other tools. mkmapdata and ncl_scripts only contain
```

```
   scripts so don't have the above build files either.

   Most tools have copies of files from other directories -- see the README.filecopies
   file for more information on this.

   Tools may also have files with the directory name followed by: namelist, runoptions,
   regional, or singlept these are sample namelists:

<directory>.namelist ------ Namelist to create a global file.
<directory>.regional ------ Namelist to create a regional file.
<directory>.singlept ------ Namelist to create a single-point file.
<directory>.runoptions ---- Command line options to use the given tool.

   These files are also used by the test scripts to test the tools (see the
   README.testing) file.

   NOTE: Be sure to change the path of the datasets references by these namelists to
   point to where you have exported your CESM inputdata datasets.

   To build:

cd <directory>
       setenv INC_NETCDF <path-to-NetCDF-include-files>
       setenv LIB_NETCDF <path-to-NetCDF-library-files>
gmake

   The process will create a file called "Depends" which has the dependencies
   for the build of each file on other files.

     By default some codes may be compiled non-optimized
     so that you can use the debugger, and with bounds-checking, and float trapping on.
     To speed up do the following...

  gmake OPT=TRUE  (by default already on for interpinic and mksurfdata_map)

     Also some of the tools allow for OpenMP shared memory parallelism
     (such as interpinic and mksurfdata) with

  gmake SMP=TRUE

   To run a program with a namelist:

       ./program < namelist

   To get help on running a program with command line options (cprnc and interpinic):

       ./program

   To run a program built with SMP=TRUE:

       setenv OMP_NUM_THREADS=<number_of_threads_to_use>

       run normally as above

III. Process sequence to create input datasets needed to run CLM

   1.) Create SCRIP grid files (if needed)

      a.) For standard resolutions these files will already be created. (done)

      b.) To create regular lat-lon regional/single-point grids run mknoocnmap.pl

       This will create both SCRIP grid files and a mapping file that will
       be valid if the region includes NO ocean whatsoever (so you can skip step 2).
       You can also use this script to create SCRIP grid files for a region
       (or even a global grid) that DOES include ocean if you use step 2 to
       create mapping files for it (simply discard the non-ocean map created by
       this script).

       Example, for single-point over Boulder Colorado.

          cd mkmapdata
          ./mknoocnmap.pl -p 40,255 -n 1x1_boulderCO

      c.) To convert from CLM or CAM grid files

       To convert CLM grid files or CAM grid files to SCRIP grid files use
       the deprecated mkgriddata program if you have to. This is also a required
```

```
     step if you want to run fine-mesh as it's the only way to get CLM topo files.

    d.) General case

     You'll need to convert or create SCRIP grid files on your own (using scripts
     or other tools) for the general case where you have an unstructured grid, or
     a grid that is not regular in latitude and longitude.

    example format
      ==================
       netcdf fv1.9x2.5_090205 {
       dimensions:
            grid_size = 13824 ;
            grid_corners = 4 ;
            grid_rank = 2 ;
       variables:
            double grid_center_lat(grid_size) ;
                    grid_center_lat:units = "degrees" ;
            double grid_center_lon(grid_size) ;
                    grid_center_lon:units = "degrees" ;
            double grid_corner_lat(grid_size, grid_corners) ;
                    grid_corner_lat:units = "degrees" ;
            double grid_corner_lon(grid_size, grid_corners) ;
                    grid_corner_lon:units = "degrees" ;
            int grid_dims(grid_rank) ;
            int grid_imask(grid_size) ;
                    grid_imask:units = "unitless" ;

  2.) Create ocean to atmosphere mapping file (if needed)

      a.) Standard resolutions (done)

      If this is a standard resolution with a standard ocean resolution -- this
      step is already done, the files already exist.

      b.) Region without Ocean (done in step 1.b)

      IF YOU RAN mknoocnmap.pl FOR A REGION WITHOUT OCEAN THIS STEP IS ALREADY DONE.

      c.) New atmosphere or ocean resolution

      If the region DOES include ocean, use gen_domain/gen_maps.sh to create a
      mapping file for it.

  Example:

  cd gen_domain
  ./gen_maps.sh -focn <ocngrid> -fatm <atmgrid> -nocn <ocnname> -natm  \
 <atmname>


  3.) Add SCRIP grid file(s) created in (1) into XML database in CLM (optional)

      See the "Adding New Resolutions or New Files to the build-namelist Database"
      Chapter in the CLM User's Guide

http://www.cesm.ucar.edu/models/cesm1.0/clm/models/lnd/clm/doc/UsersGuide/book1.html

       If you don't do this step, you'll need to specify the file to mkmapdata
       in step (3) using the "-f" option.

  4.) Create mapping files for use by mksurfdata_map with mkmapdata
      (See mkmapdata/README for more help on doing this)

     - this step uses the results of (1) that were entered into the XML database
       by step (3). If you don't enter datasets in, you need to specify the
       SCRIP grid file using the "-f" option to mkmapdata.sh.

     Example: to generate all necessary mapping files for the ne30np4 grid

         cd mkmapdata
         ./mkmapdata.sh -r ne30np4

  5.) Add mapping file(s) created in step (4) into XML database in CLM (optional)

     See notes on doing this in step (3) above.
     Edit ../bld/namelist_files/namelist_defaults_clm.xml to incorporate new
     mapping files.
```

```
   If you don't do this step, you'll need to specify the grid resolution name
   and file creation dates to mksurfdata_map in step (5) below.

6.) Convert map of ocean to atm for use by DATM and CLM with gen_domain
    (See mapping/README for more help on doing this)

   - gen_domain uses the map from step (2) (or previously created CESM maps)

   Example:

    cd ../../../../mapping/src
    gmake
    cd ..
    setenv CDATE       090206
    setenv OCNGRIDNAME gx1v6
    setenv ATMGRIDNAME fv1.9x2.5
    setenv MAPFILE $CSMDATA/cpl/cpl6/map_${OCNGRIDNAME}_to_${ATMGRIDNAME}_aave_da_${CDATE}.nc
    ./gen_domain -m $MAPFILE -o $OCNGRIDNAME -l $ATMGRIDNAME

    Normally for I compsets running CLM only you will discard the ocean domain
    file, and only use the atmosphere domain file for datm and as the fatmlndfrc
    file for CLM. Output domain files will be named according to the input OCN/LND
    gridnames.

7.) Create surface datasets with mksurfdata_map
    (See mksurfdata_map/README for more help on doing this)

   - Run mksurfdata_map/mksurfdata.pl
   - This step uses the results of step (4) entered into the XML database
     in step (5).
   - If datasets were NOT entered into the XML database, set the resolution
     to "usrspec" and use the "-usr_gname", and "-usr_gdate" options.

   Example: for 0.9x1.25 resolution

   cd mksurfdata_map/src
   gmake
   cd ..
   ./mksurfdata.pl -r 0.9x1.25

   NOTE that surface dataset will be used by default for fatmgrid - and it will
   contain the lat,lon,edges and area values for the atm grid - ASSUMING that
   the atm and land grid are the same

8.) Interpolate initial conditions using interpinic (optional)
    (See interpinic/README for more help on doing this)

9.) Add new files to XML data or using user_nl_clm (optional)

   See notes on doing this in step (3) above.

IV. Example of creating single-point datasets without entering into XML database.

   Here we apply the process described in III. for a single-point dataset
   where we don't enter the datasets into the XML database (thus skipping
   steps 3, 5 and 9), but use the needed command line options to specify where the
   files are. This also skips step (2) since step 1 creates the needed mapping file.
   We also skip step (8) and do NOT create a finidat file.

   0.) Set name of grid to use and the creation date to be used later...
      setenv GRIDNAME 1x1_boulderCO
      setenv CDATE    `date +%y%m%d`
   1.) SCRIP grid and atm to ocn mapping file
      cd mkmapdata
      ./mknoocnmap.pl -p 40,255 -n $GRIDNAME
      # Set pointer to MAPFILE that will be used in step (6)
      setenv MAPFILE `pwd`/map_${GRIDNAME}_noocean_to_${GRIDNAME}_nomask_aave_da_${CDATE}.nc
      cd ..
   2.) skip
   3.) skip
   4.) Mapping files needed for mksurfdata_map
      cd mkmapdata
      setenv GRIDFILE ../mkmapgrids/SCRIPgrid_${GRIDNAME}_nomask_${CDATE}.nc
      ./mkmapdata.sh -r $GRIDNAME -f $GRIDFILE -t regional
      cd ..
   5.) skip
   6.) Generate domain file for datm and CLM
```

```
        cd ../../../../mapping/src
        gmake
        cd ..
        setenv OCNDOM domain.ocn_noocean.nc
        setenv ATMDOM domain.lnd.{$GRIDNAME}_noocean.nc
        ./gen_domain -m $MAPFILE -o $OCNDOM -l $ATMDOM
     7.) Create surface dataset for CLM
        cd mksurfdata_map/src
        gmake
        cd ..
        ./mksurfdata.pl -r usrspec -usr_gname $GRIDNAME -usr_gdate $CDATE
     8.) skip
     9.) skip

V.  Notes on which input datasets are needed for CLM

      Fine mesh is when you are running CLM at a finer resolution than you are
      coupling it to the atmosphere. So the course resolution is the "atm" grid
      and the fine mesh is the "lnd" grid.

      global or regional/single-point grids (NO fine mesh)
        - need fsurdata and fatmlndfrc
      global grids (with fine mesh)
        - assume lat/lon grids only
          (can't use unstructured grids for either atm or lnd grid)
        - need fsurdata, fatmgrid, fatmlndfrc and fatmtopo
          (use deprecated mkgriddata to get fatmgrid, fatmlndfrc and fatmtopo)

     fsurdata ---- from mksurfdata_map in step (III.7)
     fatmlndfrc -- use the domain.lnd file from gen_domain in step (III.6)
```

# Using the cprnc tool to compare two history files

**cprnc** is a tool shared by both CAM and CLM to compare two NetCDF history files. It differences every field that has a time-axis that is also shared on both files, and reports a summary of the difference. The summary includes the three largest differences, as well as the root mean square (RMS) difference. It also gives some summary information on the field as well. You have to enter at least one file, and up to two files. With one file it gives you summary information on the file, and with two it gives you information on the differences between the two. At the end it will give you a summary of the fields compared and how many fields were different and how many were identical.

Options:

-m = do NOT align time-stamps before comparing
-v = verbose output
-ipr
-jpr
-kpr

See the **cprnc** README (../../tools/cprnc/README) file for more details.

> **Note:** To compare files with OUT a time axis you can use the **cprnc.ncl** NCL script in `models/lnd/clm/tools/ncl_scripts`. It won't give you the details on the differences but will report if the files are identical or different.

# Using interpinic to interpolate initial conditions to different resolutions

"interpinic" is used to interpolate initial conditions from one resolution to another. In order to do the interpolation you must first run CLM to create a restart file to use as the "template" to interpolate into. Running from arbitrary initial conditions (i.e. finidat = ' ') for a single time-step is sufficient to do this. Make sure the model produces a restart file. You also need to make sure that you setup the same configuration that you want to run the model with, when you create the template file.

Command line options to **interpinic**:

-i = Input filename to interpolate from
-o = Output interpolated file, and starting template file

There is a sample template file in the `models/lnd/clm/tools/interpinic` directory and can be used to run interpolate to. However, this file was created with an older version of CLM and hence we actually recommend that you would do a short run with CLM to create a template file to use.

**Example 2-1. Example of running CLM to create a template file for interpinic to interpolate to**

```
> cd scripts
> ./create_newcase -case cr_f10_TmpltI1850CN -res f10_f10 -compset I1850CN \
-mach yellowstone_intel
> cd cr_f10_TmpltI1850CN
# Set starting date to end of year, align to starting year, run a cold start, for one day
> ./xmlchange RUN_STARTDATE=1948-12-31,DATM_CLMNCEP_YR_ALIGN=1948,CLM_FORCE_COLDSTART=on,STOP_N=1
# Then setup, build and run as normal
> ./cesm_setup -case
> ./cr_f10_TmpltI1850CN.build
> ./cr_f10_TmpltI1850CN.submit
# And copy the resulting restart file to your interpinic directory
> cd ../models/lnd/clm/tools/interpinic
> cp /ptmp/$LOGIN/cr_f10_TmpltI1850CN/run/cr_f10_TmpltI1850CN.clm2.r.1949-01-01-00000.nc .
```

In the next example we build **interpinic** optimized with shared memory on for 64 threads so that it runs as fast as possible, to interpolate one of the standard 1-degree datasets to the above 10x15 template file that we created.

**Example 2-2. Example of building and running interpinic to interpolate a 1-degree `finidat` dataset to 10x15**

```
> cd models/lnd/clm/tools/interpinic
> gmake OPT=TRUE SMP=TRUE
> env OMP_NUM_THREADS=64 ./interpinic -o cr_f10_TmpltI1850CN.clm2.r.1949-01-01-00000.nc /
-i /fs/cgd/csm/inputdata/ccsm4_init/b40.1850.track1.1deg.006/0863-01-01/b40.1850.track1.1deg.006.clm2.r.0863-01-01-00000.nc
```

**Tip:** Running **interpinic** at high resolution can take a long time, so we recommend that you always build it optimized and with shared memory processing on, to cut down the run time as much as possible.

**Warning**

**interpinic** does NOT work for CNDV (bgc=cndv).

# Creating an output SCRIP grid file at a resolution to run the model on

*mkmapgrids* to create SCRIP grid data files on a grid to run the model on. The program converts old formats of CAM or CLM grid files to SCRIP grid format. There is also a NCL script (`mkscripgrid.ncl`) to create regular latitude longitude regional or single-point grids at the resolution the user desires.

SCRIP grid files for all the standard model resolutions and the raw surface datasets have already been done and the files are in the XML database. Hence, this step doesn't need to be done -- EXCEPT WHEN YOU ARE CREATING YOUR OWN GRIDS. If you have a CLM grid or CAM file from previous versions and you want to convert it you can use **mkmapgrids**.

## Using mknocnmap.pl to create grid and maps for single-point regional grids

If you want to create a regular latitude/longitude single-point or regional grid, we suggest you use **mknoocnmap.pl** in `models/lnd/clm/tools/mkmapdata` which will create both the SCRIP grid file you need (using `models/lnd/clm/tools/mkmapgrids/mkscripgrid.ncl` AND an identity mapping file assuming there is NO ocean in your grid domain. If you HAVE ocean in your domain you could modify the mask in the SCRIP grid file for ocean, and then use **ESMF_RegridWeightGen** to create the mapping file, and **gen_domain** to create the domain file. Like other tools, `mkmapdata/mknoocnmap.pl` has a help option with the following:

```
SYNOPSIS
     mknoocnmap.pl [options]     Gets map and grid files for a single land-only point.
REQUIRED OPTIONS
     -centerpoint [or -p] <lat,lon> Center latitude,longitude of the grid to create.
     -name [-or -n] <name>     Name to use to describe point

OPTIONS
     -dx <number>                   Size of total grid in degrees in longitude direction
                                    (default is 0.1)
     -dy <number>                   Size of total grid in degrees in latitude direction
                                    (default is 0.1)
     -silent [or -s]      Make output silent
     -help [or -h]      Print usage to STDOUT.
```

```
     -verbose [or -v]      Make output more verbose.
     -nx <number>                  Number of longitudes (default is 1)
     -ny <number>                  Number of latitudes  (default is 1)
ERROR: MUST set the center point
```

See Figure 2-4 for a visual representation of this process.

# Creating mapping files that mksurfdata_map will use

*mkmapdata* to create SCRIP mapping data file from SCRIP grid files (uses ESMF). The bash shell script `models/lnd/clm/tools/mkmapgrids/mkmapdata.sh` uses **ESMF_RegridWeightGen** to create a list of maps from the raw datasets that are input to **mksurfdata_map**. Each dataset that has a different grid, or land-mask needs a different mapping file for it, but many different raw datasets share the same grid/land-mask as other files. Hence, there doesn't need to be a different mapping file for EACH raw dataset -- just for each DIFFERENT raw dataset. The bash script figures out which mapping files it needs to create and then runs **ESMF_RegridWeightGen** for each one. You can then either enter the datasets into the XML database (see Chapter 3 or leave the files in place, and use the "-res usrspec -usr_gname -usr_gdate" options to **mksurfdata_map** (see the Section called *Running **mksurfdata.pl*** below). `mkmapdata.sh` has a help option with the following:

```
../../tools/mkmapdata/mkmapdata.sh

**********************
usage on bluefire:
./mkmapdata.sh

valid arguments:
[-f|--gridfile <gridname>]
     Full pathname of model SCRIP grid file to use
     This variable should be set if this is not a supported grid
     This variable will override the automatic generation of the
     filename generated from the -res argument
     the filename is generated ASSUMING that this is a supported
     grid that has entries in the file namelist_defaults_clm.xml
     the -r|--res argument MUST be specied if this argument is specified
[-r|--res <res>]
     Model output resolution (default is 10x15)
[-t|--gridtype <type>]
     Model output grid type
     supported values are [regional,global], (default is global)
[-b|--batch]
     Toggles batch mode usage. If you want to run in batch mode
     you need to have a separate batch script for a supported machine
     that calls this script interactively - you cannot submit this
     script directory to the batch system
[-l|--list]
     List mapping files required (use check_input_data to get them)
     also writes data to clm.input_data_list
[-d|--debug]
     Toggles debug-only (don't actually run mkmapdata just echo what would happen)
[-o|--ocn] <ogrid>
     Also map to the input ocean-grid resolution
[-h|--help]
     Displays this help message
[-v|--verbose]
     Toggle verbose usage -- log more information on what is happening

 You can also set the following env variables:
  ESMFBIN_PATH - Path to ESMF binaries
                 (default is /contrib/esmf-5.3.0-64-O/bin)
  CSMDATA ------ Path to CESM input data
                 (default is /glade/proj3/cseg/inputdata)
```

```
  MPIEXEC ------ Name of mpirun executable
                 (default is mpirun.lsf)
  REGRID_PROC -- Number of MPI processors to use
                 (default is 8)

**pass environment variables by preceding above commands
  with 'env var1=setting var2=setting '
**********************
```

# Creating a domain file for CLM and DATM

*gen_domain* to create a domain file for datm from a mapping file. The domai file is then used by BOTH datm AND CLM to define the grid and land-mask. The general data flow is shown in two figures. Figure 2-3 shows the general flow for a general global case (or for a regional grid that DOES include ocean). Figure 2-4 shows the use of **mknoocnmap.pl** (see the Section called *Using **mknocnmap.pl** to create grid and maps for single-point regional grids*) to create a regional or single-point map file that is then run through **gen_domain** to create the domain file for it. As stated before Figure 2-2 is the legend for both of these figures. See the `mapping/gen_domain_files/README` (../../../../../mapping/gen_domain_files/README) file for more help on **gen_domain**.

Here we create domain files for a regular global domain.

**Figure 2-3. Global Domain file creation**

Starting from SCRIP grid files for both your atmosphere and ocean, you use **gen_cesm_maps.sh** to create a mapping file between the atmosphere and ocean. That mapping file is then used as input to **gen_domain** to create output domain files for both atmosphere and ocean. The atmosphere domain file is then used by both CLM and DATM for I compsets, while the ocean domain file is ignored. For this process you have to define your SCRIP grid files on your own. For a regional or single-point case that doesn't include ocean see Figure 2-4. (See Figure 2-2 for the legend for this figure)

Note, that the SCRIP grid file used to start this process, is also used in **mkmapdata.sh** (see the Section called *Creating mapping files that **mksurfdata_map** will use*). Next we create domain files for a single-point or regional domain.

**Figure 2-4. Domain file creation using mknoocnmap.pl**

For a regular latitude/longitude grid that can be used for regional or single point simulations -- you can use **mknoocnmap.pl**. It creates a SCRIP grid file that can then be used as input to **mkmapdata.sh** as well as a SCRIP mapping file that is then input to **gen_domain**. The output of **gen_domain** is a atmosphere domain file used by both CLM and DATM and a ocean domain file that is ignored. (See Figure 2-2 for the legend for this figure)

In this case the process creates both SCRIP grid files to be used by **mkmapdata.sh** as well as the domain files that will be used by both CLM and DATM.

# Creating a set of regional datasets from existing global datasets

Use the *ncl_scripts/getregional_datasets.pl* script to extract a region or a single-point from global input datasets. See the Section called *Creating a set of regional datasets from existing global datasets* in the single-point chapter for more information on this.

# Using mksurfdata_map to create surface datasets from grid datasets

**mksurfdata_map** is used to create surface-datasets from grid datasets and raw datafiles at half-degree resolution to produce files that describe the surface characteristics needed by CLM (fraction of grid cell covered by different land-unit types, and fraction for different vegetation types, as well as things like soil color, and soil texture, etc.). To run **mksurfdata_map** you can either use the **mksurfdata.pl** script which will create namelists for you using the **build-namelist** XML database, or you can run it by hand using a namelist that you provide (possibly modeled after an example provided in the `models/lnd/clm/tools/mksurfdata_map` directory). The namelist for **mksurfdata_map** is sufficiently complex that we recommend using the **mksurfdata.pl** tool to build them. It also requires that mapping files from your output grid to the raw datasets that **mksurfdata_map** are made to regrid relevant datasets (see the Section called *Creating mapping files that **mksurfdata_map** will use* and Figure 2-1 for a visual representation of the process). For standard resolutions these mapping files are already created, but if you want to run for your own single-point or region, you'll need to create these mapping files. In the next section we describe how to use the **mksurfdata.pl** script and the following section gives more details on running **mksurfdata_map** by hand and the various namelist input variables to it.

## Running mksurfdata.pl

The script **mksurfdata.pl** can be used to run the **mksurfdata_map** program for several configurations, resolutions, simulation-years and simulation year ranges. It will create the needed namelists for you and move the files over to your inputdata directory location (and create a list of the files created, and for developers this file is also a script to import the files into the svn inputdata repository). It will also use the **build-namelist** XML database to determine the correct input files to use, and for transient cases it will create the appropriate `mksrf_fdynuse` file with the list of files for each year needed for this case. And

in the case of urban single-point datasets (where surface datasets are actually input into **mksurfdata_map**) it will do the additional processing required so that the output dataset can be used once again by **mksurfdata_map**. Because, it figures out namelist and input files for you, it is recommended that you use this script for creation of standard surface datasets. If you need to create surface datasets for customized cases, you might need to run **mksurfdata_map** on it's own. But you could use **mksurfdata.pl**with the "-debug" option to give you a namelist to start from. The list of files needed is very long and not necessisarily easy to figure out. For help on **mksurfdata.pl** you can use the "-help" option as below:

```
> cd models/lnd/clm/tools/mksurfdata_map
> ./mksurdata.pl -help
```

The output of the above command is:

```
SYNOPSIS

     For supported resolutions:
     mksurfdata.pl -res <res>  [OPTIONS]
        -res [or -r] is the supported resolution(s) to use for files (by default all ).


     For unsupported, user-specified resolutions:
     mksurfdata.pl -res usrspec -usr_gname <user_gname> -usr_gdate <user_gdate>   \
   [OPTIONS]
        -usr_gname "user_gname"    User resolution name to find grid file with
                                   (only used if -res is set to 'usrspec')
        -usr_gdate "user_gdate"    User map date to find mapping files with
                                   (only used if -res is set to 'usrspec')
                                   NOTE: all mapping files are assumed to be in mkmapdata
                                    - and the user needs to have invoked mkmapdata in
                                      that directory first

OPTIONS
     -allownofile                  Allow the script to run even if one of the input files
                                   does NOT exist.
     -crop                         Add in crop datasets
     -dinlc [or -l]                Enter the directory location for inputdata
                                   (default /glade/proj3/cseg/inputdata)
     -debug [or -d]                Do not actually run -- just print out what
                                   would happen if ran.
     -dynpft "filename"            Dynamic PFT/harvesting file to use
                                   (rather than create it on the fly)
                                   (must be consistent with first year)
     -glc_nec "number"             Number of glacier elevation classes to use (by default 0)
     -hires                        If you want to use high-resolution input datasets rather
                                   than the default lower resolution datasets
                                   (low resolution is typically at half-degree)
     -irrig                        If you want to include irrigated crop in the output file.
     -exedir "directory"           Directory where mksurfdata_map program is
                                   (by default assume it is in the current directory)
     -mv                           If you want to move the files after creation to the
                                   correct location in inputdata
                                   (by default -nomv is assumed so files are NOT moved)
     -new_woodharv                 Use the new good wood harvesting (for rcp6 and rcp8.5)
                                   (by default use the old harvesting used for IPCC simulations)
     -years [or -y]                Simulation year(s) to run over (by default 1850,2000)
                                   (can also be a simulation year range: i.e. 1850-2000)
     -help  [or -h]                Display this help.

     -rcp   [or -c] "rep-con-path" Representative concentration pathway(s) to use for
                                   future scenarios
                                   (by default -999.9, where -999.9 means historical ).
     -usrname "clm_usrdat_name"    CLM user data name to find grid file with.

      NOTE: years, res, and rcp can be comma delimited lists.


OPTIONS to override the mapping of the input gridded data with hardcoded input
```

```
-pft_frc "list of fractions"  Comma delimited list of percentages for veg types
-pft_idx "list of veg index"  Comma delimited veg index for each fraction
-soil_cly "% of clay"         % of soil that is clay
-soil_col "soil color"        Soil color (1 [light] to 20 [dark])
-soil_fmx "soil fmax"         Soil maximum saturated fraction (0-1)
-soil_snd "% of sand"         % of soil that is sand
```

To run the script with optimized **mksurfdata_map** for a 4x5 degree grid for 1850 conditions, on yellowstone you would do the following:

**Example 2-3. Example of running mksurfdata.pl to create a 4x5 resolution `fsurdat` for a 1850 simulation year**

```
> cd models/lnd/clm/tools/mksurfdata_map
> gmake USER_FC=ifort
> ./mksurfdata.pl -y 1850 -r 4x5
```

# Running mksurfdata_map by Hand

In the above section we show how to run **mksurfdata_map** through the **mksurfdata.pl** using input datasets that are in the **build-namelist** XML database. When you are running with input datasets that are NOT available in the XML database you either need to add them as outlined in Chapter 3, or you need to run **mksurfdata_map** by hand, as we will outline here. The easiest way to start is to use the "-debug" option to **mksurfdata.pl** for a case as close as possible and then customize the resulting `namelist` file for the datasets that you change.

## Preparing your mksurfdata_map namelist

When running **mksurfdata_map** by hand you will need to prepare your own input namelist. There is a sample namelist setup for running on the previous NCAR machine bluefire. So you will need to change the filepaths to use that namelist. The sample namelist is called

`mksurfdata_map.namelist` -- standard sample namelist.
`pftdyn_hist_simyr1850-2005.txt` -- the `mksrf_fdynuse` text file with filenames for 1850-2005.

Note, that one of the inputs `mksrf_fdynuse` is a filename that includes the filepaths to other files. The filepaths in this file will have to be changed as well. You also need to make sure that the line lengths remain the same as the read is a formatted read, so the placement of the year in the file, must remain the same, even with the new filenames. One advantage of the **mksurfdata.pl** script is that it will create the `mksrf_fdynuse` file for you.

We list the namelist items below. Most of the namelist items are filepaths to give to the input half degree resolution datasets that you will use to scale from to the resolution of your grid dataset. You must first specify the input grid dataset for the resolution to output for:

1. `mksrf_fgrid` mapping file that defines the output grid to run on

Then you must specify settings for input high resolution datafiles

1. `mksrf_ffrac` land fraction and land mask dataset
2. `mksrf_fglacier` Glacier dataset
3. `mksrf_flai` Leaf Area Index dataset
4. `mksrf_flanwat` Land water dataset
5. `mksrf_forganic` Organic soil carbon dataset
6. `mksrf_fmax` Max fractional saturated area dataset
7. `mksrf_fsoicol` Soil color dataset
8. `mksrf_fsoitex` Soil texture dataset
9. `mksrf_ftopo` Topography dataset (this is used to limit the extent of urban regions and is used for glacier multiple elevation classes)
10. `mksrf_furban` Urban dataset
11. `mksrf_fvegtyp` PFT vegetation type dataset
12. `mksrf_fvocef` Volatile Organic Compound Emission Factor dataset

Then the list of mapping files for each of these datasets. The same mapping file can be used by multiple raw datasets (from the list above) if they are on the same grid and land-mask. Each mapping file needs to correspond to the grid and land-mask of the raw datasets above.

1. `map_fglacier` mapping file for `mksrf_fglacier`
2. `map_flai` mapping file for `mksrf_flai`
3. `map_flakwat` mapping file for `mksrf_flakwat`
4. `map_forganic` mapping file for `mksrf_forganic`
5. `map_fmax` mapping file for `mksrf_fmax`
6. `map_fsoicol` mapping file for `mksrf_fsoicol`
7. `map_fsoitex` mapping file for `mksrf_fsoitex`
8. `map_furbtopo` mapping file for `mksrf_furbtopo`
9. `map_flndtopo` mapping file for `mksrf_flndtopo`
10. `map_fharvest` mapping file for `mksrf_fharvest`
11. `map_fwetlnd` mapping file for `mksrf_fwetlnd`
12. `map_furban` mapping file for `mksrf_furban`
13. `map_fpft` mapping file for `mksrf_fpft`
14. `map_fvocef` mapping file for `mksrf_fvocef`

**Note:** If you add new raw datasets to **mksurfdata_map**, you will need to add the corresponding mapping file for that dataset as well. If the file is on the same grid and land-mask as another dataset, it can share the same mapping file. If it is on a different grid and/or land-mask -- YOU WILL NEED

TO CREATE MAPPING DATASETS FOR IT. And **mkmapdata.sh** will need to be changed to create the new mapping files (see the Section called *Creating mapping files that **mksurfdata_map** will use*. See Figure 2-5 for a visual representation of the relationship of the various input and output files for **mksurfdata_map**.

**Figure 2-5. Details of running mksurfdata_map**

Each of the raw datasets (the mksrf_* files) needs a mapping file to map from the output grid you are running on to the grid and land-mask for that dataset. Some raw datasets share the same grid and land mask -- hence they can share the same mapping file. One of the mapping files is used to specify the grid for mksrf_fgrid.

You specify the ASCII text file with the land-use files.

1. `mksrf_fdynuse` "dynamic land use" for transient land-use/land-cover changes. This is an ASCII text file that lists the filepaths to files for each year and then the year it represents (note: you MUST change the filepaths inside the file when running on a machine NOT at NCAR). We always use this file, even for creating datasets of a fixed year. Also note that when using the "pft_" settings this file will be an XML-like file with settings for PFT's rather than filepaths (see the Section called *Single Point options to* ***mksurfdata_map*** below).

And optionally you can specify settings for:

1. `all_urban` If entire area is urban (typically used for single-point urban datasets, that you want to be exclusively urban)
2. `mksrf_firrig` Irrigation dataset, if you want activate the irrigation model over generic cropland (experimental mode, normally NOT used). If this dataset is set, you also NEED to set the mapping file for the irrigation dataset with `map_firrig`.
3. `mksrf_gridnm` Name of output grid resolution (if not set the files will be named according to the number of longitudes by latitudes)
4. `mksrf_gridtype` Type of grid (default is 'global')
5. `nglcec` number of glacier multiple elevation classes. Can be 0, 1, 3, 5, or 10. When using the resulting dataset with CLM you can then run with `glc_nec` of either 0 or this value. (experimental normally use the default of 0, when running with the land-ice model in practice only 10 has been used)
6. `numpft` number of Plant Function Types (PFT) in the input vegetation `mksrf_fvegtyp` dataset. You change this to 20, if you want to create a dataset with prognostic crop activated. The vegetation dataset also needs to have prognostic crop types on it as well. (experimental normally not changed from the default of 16)
7. `outnc_large_files` If output should be in NetCDF large file format
8. `outnc_double` If output should be in double precision (normally we turn this on)
9. `pft_frc` array of fractions to override PFT data with for all gridpoints (experimental mode, normally NOT used).
10. `pft_idx` array of PFT indices to override PFT data with for all gridpoints (experimental mode, normally NOT used).
11. `soil_clay` percent clay soil to override all gridpoints with (experimental mode, normally NOT used).
12. `soil_color` Soil color to override all gridpoints with (experimental mode, normally NOT used).

13. `soil_fmax` Soil maximum fraction to override all gridpoints with (experimental mode, normally NOT used).
14. `soil_sand` percent sandy soil to override all gridpoints with (experimental mode, normally NOT used).

After creating your namelist, when running on a non NCAR machine you will need to get the files from the inputdata repository. In order to retrieve the files needed for mksurfdata_map you can do the following on your namelist to get the files from the inputdata repository, using the **check_input_data** script which also allows you to export data to your local disk.

**Example 2-4. Getting the raw datasets for mksurfdata_map to your local machine using the check_input_data script**

```
> cd models/lnd/clm/tools/mksurfdata_map
# First remove any quotes and copy into a filename that can be read by the
# check_input_data script
> sed "s/'//g" namelist > clm.input_data_list
# Run the script with -export and give the location of your inputdata with $CSMDATA
> ../../../../../scripts/ccsm_utils/Tools/check_input_data -datalistdir . \
-inputdata $CSMDATA -check -export
# You must then do the same with the fdynuse file referred to in the namelist
# in this case we add a file = to the beginning of each line
> awk '{print "file = "$1}' pftdyn_hist_simyr2000-2000.txt > clm.input_data_list
# Run the script with -export and give the location of your inputdata with $CSMDATA
> ../../../../../scripts/ccsm_utils/Tools/check_input_data -datalistdir . \
-inputdata $CSMDATA -check -export
```

### *Single Point options to **mksurfdata_map***

The options: pft_frc, pft_idx, soil_clay, soil_color, soil_fmax, and soil_sand exist to override the values that come in on the datasets with user specified values. They override the PFT and soil values for all grid points to the given values that you set. This is useful for running with single-point tower sites where the soil type and vegetation is known. Note that when you use pft_frc, all other landunits will be zeroed out, and the sum of your pft_frc array MUST equal 100.0. Also note that when using the "pft_" options the `mksrf_fdynuse` file instead of having filepath's will be an XML-like file with PFT settings. Unlike the file of file-paths, you will have to create this file by hand, **mksurfdata.pl** will NOT be able to create it for you (other than the first year which will be set to the values entered on the command line). Instead of a filepath you have a list of XML elements that give information on the PFT's and harvesting for example:

```
<pft_f>100</pft_f><pft_i>1</pft_i><harv>0,0,0,0,0</harv><graz>0</graz>
```

So the <pft_f> tags give the PFT fractions and the <pft_i> tags give the index for that fraction. Harvest is an array of five elements, and grazing is a single value. Like the usual file each list of XML elements goes with a year, and there is limit on the number of characters that can be used.

## Standard Practices when using mksurfdata_map

In this section we give the recommendations for how to use **mksurfdata_map** to give similar results to the files that we created when using it.

If you look at the standard surface datasets that we have created and provided for use, there are three practices that we have consistently done in each (you also see these in the sample namelists and in the **mksurfdata.pl** script). The first is that we always output data in double precision (hence `outnc_double` is set to `.true.`). The next is that we always use the procedure for creating transient datasets (using `mksrf_fdynuse`) even when creating datasets for a fixed simulation year. This is to ensure that the fixed year datasets will be consistent with the transient datasets. When this is done a "surfdata.pftdyn" dataset will be created -- but will NOT be used in CLM. If you look at the sample namelist `mksurfdata_map.namelist` you note that it sets `mksrf_fdynuse` to the file `pftdyn_hist_simyr2000.txt`, where the single file entered is the same PFT file used in the rest of the namelist (as `mksrf_fvegtyp`). The last practice that we always do is to always set `mksrf_ftopo`, even if glacier elevation classes are NOT active. This is important in limiting urban areas based on topographic height, and hence is important to use all the time. The glacier multiple elevation classes will be used as well if you are running a compset with the active glacier model.

There are two other important practices for creating urban single point datasets. The first is that you often will want to set `all_urban` to `.true.` so that the dataset will have 100% of the gridcell output as urban rather than some mix of: urban, vegetation types, and other landunits. The next practice is that most of our specialized urban datasets have custom values for the urban parameters, hence we do NOT want to use the global urban dataset to get urban parameters -- we use a previous version of the surface dataset for the urban parameters. However, in order to do this, we need to append onto the previous surface dataset the grid and land mask/land fraction information from the grid and fraction datasets. This is done in **mksurfdata.pl** using the NCO program **ncks**. An example of doing this for the Mexico City, Mexico urban surface dataset is as follows:

```
> ncks -A $CSMDATA/lnd/clm2/griddata/griddata_1x1pt_mexicocityMEX_c090715.nc \
$CSMDATA/lnd/clm2/surfdata/surfdata_1x1_mexicocityMEX_simyr2000_c100407.nc
> ncks -A $CSMDATA/lnd/clm2/griddata/fracdata_1x1pt_mexicocityMEX_navy_c090715.nc \
$CSMDATA/lnd/clm2/surfdata/surfdata_1x1_mexicocityMEX_simyr2000_c100407.nc
```

Note, if you look at the current single point urban surface datasets you will note that the above has already been done.

The final issue is how to build **mksurfdata_map**. When NOT optimized **mksurfdata_map** is a bit slower, but not too bad. Previous versions of **mksurfdata_map** were much slower because it calculated the mapping between each grid on the fly. Now, the mapping is done outside **mksurfdata_map** using the ESMF regridding tools. So you may want to run it optimized, which is the default. The problem with running optimized is that answers will be different when running optimized versus non-optimized for most compilers. So if you want answers to be the same as a previous surface dataset, you will need to run it on the same platform and optimization level. Note, that the output surface datasets will have attributes that describe whether the file was written out optimized or not, to enable the user to more easily try to match datasets created previously. For more information on the different compiler options for the CLM4 tools see the Section called *Common environment variables and options used in building the FORTRAN tools*.

# Converting unstructured grid output to gridded datasets for post processing

*mkprocdata_map* to interpolate output unstructured grids (such as the CAM HOMME dy-core "ne" grids like ne30np4) into a 2D regular lat/long grid format that can be plotted easily. It does this by using the ESMF regrid mapping utility to map between the model grid and a standard latitude-longitude grid of your choosing. See the README (../../tools/mkprocdata_map/README) file for more information on how to run this program.

# How to Customize Datasets for particular Observational Sites

There are two ways to customize datasets for a particular observational site. The first is to customize the input to the tools that create the dataset, and the second is to over-write the default data after you've created a given dataset. Depending on the tool it might be easier to do it one way or the other. In Table 3-1 we list the files that are most likely to be customized and the way they might be customized. Of those files, the ones you are most likely to customize are: fatmlndfrc, fsurdat, faerdep (for DATM), and stream_fldfilename_ndep. Note **mksurfdata_map** as documented previously has options to overwrite the vegetation and soil types. For more information on this also see the Section called *Creating your own single-point/regional surface datasets* in Chapter 5.

Another aspect of customizing your input datasets is customizing the input atmospheric forcing datasets. See the the Section called *Running with your own atmosphere forcing* in Chapter 5 for more information on this.

# Conclusion of tools description

We've given a description of how to use the different tools with CLM to create customized datasets. In the next chapter we will talk about how to make these files available for build-namelist so that you can easily create simulations that include them. In the chapter on single-point and regional datasets we also give an alternative way to enter new datasets without having to edit files.

# Chapter 3. Adding New Resolutions or New Files to the build-namelist Database

In the last chapter we gave the details on how to create new files for input into CLM. These files could be either global resolutions, regional-grids or even a single grid point. If you want to easily have these files available for continued use in your development you will then want to include them in the build-namelist database so that build-namelist can easily find them for you. You can deal with them, just by putting the settings in the `user_nl_clm` namelist file, or by using CLM_USRDAT_NAME. Another way to deal with them is to enter them into the database for build-namelist, so that build-namelist can find them for you. This keeps one central database for all your files, rather than having multiple locations to keep track of files. If you have a LOT of files to keep track of it also might be easier than keeping track by hand, especially if you have to periodically update your files. If you just have a few quick experiments to try, for a short time period you might be best off using the other methods mentioned above.

There are two parts to adding files to the build-namelist database. The first part is adding new resolution names which is done in the `models/lnd/clm/bld/namelist_files/namelist_definition.xml` file (and in the `models/lnd/clm/bld/config_files/config_definition.xml` file when adding supported single-point datasets). You can then use the new resolution by using CLM_USRDAT_NAME. If you also want to be able to give the resolution into **create_newcase** -- you'll need to add the grid to the `scripts/ccsm_utils/Case.template/config_grid.xml` file.

The second part is actually adding the new filenames which is done in the `models/lnd/clm/bld/namelist_files/namelist_defaults_clm.xml` file (`models/lnd/clm/bld/namelist_files/namelist_defaults_clm_tools.xml` file for CLM tools). If you aren't adding any new resolutions, and you are just changing the files for existing resolutions, you don't need to edit the namelist_definition file.

## Managing Your Own Data-files

If you are running on a supported machine (such as yellowstone or hopper) the standard input datasets will already be available and you won't have to check them out of the subversion inputdata server. However, you also will NOT be able to add your own datafiles to these standard inputdata directories -- because most likely you won't have permissions to do so. In order to add files to the XML database or to use CLM_USRDAT_NAME you need to put data in the standard locations so that they can be found. The recommended way to do this is to use the **link_dirtree** tool in the CESM scripts. Some information on **link_dirtree** is available in the CESM1.1.1 Scripts User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm). We also have some examples of it's use here and in other sections of this User's Guide.

Using **link_dirtree** is quite simple, you give the directory where data exists and then the directory that you want to create where datasets will point to the original source files. In the example below we use "$HOME/inputdata", but MYCSMDATA could be any directory you have access to where you want to put your data.

```
> cd scripts
# First make sure you have a inputdata location that you can write to
```

```
# You only need to do this step once, so you won't need to do this in the future
# (except to bring in any updated files in the original $CSMDATA location).
> setenv MYCSMDATA $HOME/inputdata    # Set env var for the directory for input data
> ./link_dirtree $CSMDATA $MYCSMDATA
```

Then when you create a case you will change DIN_LOC_ROOT_CSMDATA to point to the location you linked to rather than the default location.

```
> ./xmlchange DIN_LOC_ROOT_CSMDATA=$MYCSMDATA
```

In order to list the files that you have created you merely need to use the UNIX command **find** to find the files that are NOT softlinks. So for example executing the following command:

```
> find $MYCSMDATA -type f -print
```

for me gives the following list of CLM_USRDAT_NAME files that I have created.

```
/blhome/erik/inputdata/atm/cam/chem/trop_mozart_aero/aero/aerosoldep_monthly_1849-2006_1x
/blhome/erik/inputdata/atm/cam/chem/trop_mozart_aero/aero/aerosoldep_monthly_1849-2006_13
/blhome/erik/inputdata/atm/cam/chem/trop_mozart_aero/aero/aerosoldep_rcp8.5_monthly_1850-
/blhome/erik/inputdata/atm/cam/chem/trop_mozart_aero/aero/aerosoldep_rcp4.5_monthly_1850-
/blhome/erik/inputdata/atm/datm7/domain.clm/domain.lnd.1x1pt_US-Ha1_USGS.nc
/blhome/erik/inputdata/atm/datm7/domain.clm/domain.lnd.13x12pt_f19_alaskaUSA_gx1v6.nc
/blhome/erik/inputdata/lnd/clm2/griddata/fracdata_13x12pt_f19_alaskaUSA_gx1v6.nc
/blhome/erik/inputdata/lnd/clm2/griddata/fracdata_1x1pt_US-Ha1_USGS.nc
/blhome/erik/inputdata/lnd/clm2/griddata/topodata_13x12pt_f19_alaskaUSA.nc
/blhome/erik/inputdata/lnd/clm2/griddata/griddata_1x1pt_US-Ha1.nc
/blhome/erik/inputdata/lnd/clm2/griddata/griddata_13x12pt_f19_alaskaUSA.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata_13x12pt_f19_alaskaUSA_simyr1850.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata_1x1pt_US-Ha1_simyr2000.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_rcp4.5_13x12pt_f19_alaskaUSA_sim
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata_1x1pt_US-Ha1_simyr1850.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata_13x12pt_f19_alaskaUSA_simyr2000.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_1x1pt_US-Ha1_simyr1849-2006.nc
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_13x12pt_f19_alaskaUSA_simyr1850-
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_rcp8.5_13x12pt_f19_alaskaUSA_sim
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_13x12pt_f19_alaskaUSA_simyr1849-
/blhome/erik/inputdata/lnd/clm2/surfdata/surfdata.pftdyn_1x1pt_US-Ha1_simyr1850-2100.nc
```

You can also use **find** to list files that have a particular pattern in the name as well (using the -name option with wildcards). Also you can always rerun the **link_dirtree** command if any new files are added that you need to be linked into your directory tree. Since, the files are soft-links -- it doesn't take up much space other than the files that you add there. This way all of the files are kept in one place, they are

organized by usage according to CESM standards, and you can easily find your own files, and CLM can find them as well.

# Adding Resolution Names

If you are adding files for new resolutions which aren't covered in the namelist_definition file -- you'll need to add them in. The list of valid resolutions is in the id="res" entry in the `models/lnd/clm/bld/namelist_files/namelist_definition.xml` file. You need to choose a name for your new resolution and simply add it to the comma delimited list of valid_values for the id="res" entry. The convention for global Gaussian grids is number_of_latitudes x number_of_longitudes. The convention for global finite volume grids is latitude_grid_size x longitude_grid_size where latitude and longitude is measured in degrees. For regional or single-point datasets the names have a grid size number_of_latitudes x number_of_longitudes followed by an underscore and then a descriptive name such as a City name followed by an abbreviation for the Country in caps. The only hard requirement is that names be unique for different grid files. Here's what the entry for resolutions looks like in the file:

```
<entry id="res" type="char*30" category="default_settings"
       group="default_settings"
       valid_values=
"128x256,64x128,48x96,32x64,8x16,94x192,0.23x0.31,0.47x0.63,
0.9x1.25,1.9x2.5,2.65x3.33,4x5,10x15,5x5_amazon,1x1_tropicAtl,
1x1_camdenNJ,1x1_vancouverCAN,1x1_mexicocityMEX,1x1_asphaltjungleNJ,
1x1_brazil,1x1_urbanc_alpha,0.5x0.5">
Horizontal resolutions
</entry>
```

As you can see you just add your new resolution names to the end of the valid_values list.

When using PTCLM and adding supported single-point resolutions, you'll also want to add these resolutions to the `models/lnd/clm/bld/config_files/config_definition.xml` under the sitespf_pt name. The entry in that file looks like:

```
<entry id="sitespf_pt"
valid_values="none,1x1_brazil,1x1_tropicAtl,5x5_amazon,
1x1_camdenNJ,1x1_vancouverCAN,1x1_mexicocityMEX,1x1_asphaltjungleNJ,
1x1_urbanc_alpha,1x1_numaIA,1x1_smallvilleIA"
value="none" category="physics">
Flag to turn on site specific special configuration flags for supported single
point resolutions.
Currently the only special settings are for MEXICOCITY and VANCOUVER, which make
changes to urban parameters.
</entry>
```

PTCLM assumes that any supported single-point resolutions are valid settings for sitespf_pt.

# Adding or Changing Default Filenames

To add or change the default filenames you edit the `models/lnd/clm/bld/namelist_files/namelist_defaults_clm.xml` and either change an existing filename or add a new one. Most entries in the default namelist files, include different attributes

that describe the different properties that describe the differences in the datasets. Attributes include the: resolution, year to simulation, range of years to simulate for transient datafiles, the land-mask, the representative concentration pathway (rcp) for future scenarios, and the type of biogeochemistry (bgc) model used. For example the `fatmgrid` for the 1.9x2.5 resolution is as follows:

```
<fatmgrid hgrid="1.9x2.5" >lnd/clm2/griddata/griddata_1.9x2.5_060404.nc
</fatmgrid>
```

Other `fatmgrid` files are distinguished from this one by their resolution (hgrid) attribute.

To add or change the default filenames for CLM tools edit the `models/lnd/clm/bld/namelist_files/namelist_defaults_clm_tools.xml` and either change an existing filename or add a new one. Editing this file is similar to the `namelist_defaults_clm.xml` talked about above.

## What are the required files?

Different types of simulations and different types of configurations for CLM require different lists of files. The Carbon Nitrogen (cn) Biogeochemistry model for example requires `stream_fldfilename_ndep` files, which are NOT required by other bgc modes. Transient simulations also require transient datasets, and the names of these datasets are sometimes different from the static versions (sometimes both are required as in the dynamic PFT cases).

In the following table we list the different files used by CLM, they are listed in order of importance, dependencies, and customizing. So the required files are all near the top, and the files used only under different conditions are listed later, and files with the fewest dependencies are near the top, as are the files that are least likely to be customized.

**Table 3-1. Required Files for Different Configurations and Simulation Types**

| Filename | Config. type | Simulation type | Resol. Dependent? | Other Dependencies |
|---|---|---|---|---|
| | **Notes** | | | |
| fpftcon | ALL | ALL | No | No |
| | Not usually customized, as describes plant function type properties. | | | |
| fsnowoptics | ALL | ALL | No | No |
| | Not usually customized as describes global snow optical properties. | | | |
| fsnowaging | ALL | ALL | No | No |
| | Not usually customized as describes global snow aging properties. | | | |
| fatmgrid | ALL | ALL | Yes | No |

| Filename | Config. type | Simulation type | Resol. Dependent? | Other Dependencies |
|---|---|---|---|---|
| | **Notes**<br>Creating, using **mkgriddata** usually gives you the amount of customization you need, as it just describes the grid and grid extents. | | | |
| fatmlndfrc | ALL | ALL | Yes | land-mask |
| | Describes the land-mask for points with active land, as well as the fraction of each grid-cell covered by land. You might customize it to make sure the land-fraction of your grid-cell matches the expected values for your site. But, usually you will just use what gen_domain gives you (this can be in the CLM only format, or the CESM domain file format the CLM only format is deprecated). | | | |
| fsurdat | ALL | ALL | Yes | simulation-year |
| | Describes percentages of different land-units, columns and vegetation types within each grid-cell. To customize for a specific point or region you may want to use custom input datasets to **mksurfdata_map** when creating the file. **mksurfdata_map** also allows you to customize the PFT, and soil types to it see the Section called *Single Point options to **mksurfdata_map*** in Chapter 2. | | | |
| fpftdyn | ALL | transient land-use land-cover change | Yes | Simulation year range, and representative concentration pathway (rcp) |
| | See notes on fsurdat files. | | | |
| frivinp_rtm | RTM only | ALL | No | No |
| | We only provide a half-degree global river routing file. If you want to model river flow for a smaller scale, or a basin regional scale, you would need to create your own custom file to do that. Normally, we turn river-routing OFF for regional or single point simulations. | | | |
| finidat | ALL | RUN_TYPE="startup", CLM_FORCE_COLDSTART="off" | Yes | mask, maxpft, bg simulation-year, start-date |
| | Used for starting the model from a spun-up state. Create these files by running the model for multiple years and saving the restart file from the end of a spin-up simulation. Or use the CLM tool **interpinic** to interpolate initial conditions. | | | |
| fglcmask | glc_nec > 0 | Used for simulations with the active glacier ice sheet model "cism" | Yes | glacier-grid |

| Filename | Config. type | Simulation type | Resol. Dependent? | Other Dependencies |
|---|---|---|---|---|
| | **Notes**<br>Needs to match the file used by "cism" and be for the same glacier grid. Only customized as coupled with the glacier model. | | | |
| stream_fldfilename_ndep | bgc=cn/cndv | Yes | No | simulation-year |
| | You may customize this file to get the Nitrogen deposition characteristics of your site if available. This file will be interpolated while the model is running from it's resolution to the resolution that CLM is running at. | | | |

# Chapter 4. How to run some special cases

In this chapter we describe how to run some special cases that take more than one step to do. The straightforward cases have compsets and/or build-namelist use-cases setup for them or require simple editing of a single-case. All of the cases here require you to do at least two simulations with different configurations, or require more complex editing of the case (changing the streams files).

The nine cases we will describe are:

1. *Running with the prognostic crop model on*
2. *Running with the irrigation model on*
3. *Spinning up the Satellite Phenology Model (CLMSP spinup)*
4. *Spinning up the biogeochemistry Carbon-Nitrogen Model (CN spinup)*
5. *Spinning up the Carbon-Nitrogen Dynamic Global Vegetation Model (CNDV spinup)*
6. *Running with MOAR data as atmospheric forcing to spinup the model*
7. *Running with your own previous simulation as atmospheric forcing to spinup the model*
8. *Doing perturbation error growth tests*
9. *Running stand-alone CLM with transient historical $CO_2$ concentration*

**Caution**

The cases in this chapter are more sophisticated and require more technical knowledge and skill than cases in previous chapters. The user should be very familiar with doing simple cases before moving onto the cases described here.

## Running with the prognostic crop model on

In CLM4 in CESM1.0.3 a prognostic crop model was added to CLM4. The prognostic crop model is setup to work with CN for present day conditions and we have surface and initial condition datasets at f19 resolution. In order to use the initial condition file, we need to set the RUN_TYPE to `startup` rather than `hybrid` since the compset for f19 sets up to use an initial condition file without crop active. To activate the crop model we simply add "-crop on" to CLM_CONFIG_OPTS.

**Example 4-1. Example Crop Simulation**

```
> cd scripts
> ./create_newcase -case CROP -res f19_g16 -compset ICN -mach yellowstone_intel
> cd CROP
# Append "-crop on" to CLM_CONFIG_OPTS in env_build.xml (you could also use an editor)
> ./xmlchange CLM_CONFIG_OPTS="-crop on" -append
# Change to startup type so uses spunup initial conditions file for crop if it exists
# By default the model will do a hybrid startup with an initial condition file
# incompatible with the crop surface dataset.
> ./xmlchange RUN_TYPE=startup
> ./cesm_setup
# Now build and run normally
```

```
> ./CROP.build
> ./CROP.submit
```

# Running with the irrigation model on

In CLM4 in CESM1.0.3 an irrigation model for generic crop was added to CLM4. Currently, irrigation and crop can NOT be used together see bug number 1326 in the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file. The irrigation model is tuned to work only with CLMSP see the caution below for for more information on this. To turn on irrigation we simply add "-irrig on" to CLM_BLDNML_OPTS. Just as in the crop example we also change RUN_TYPE to `startup` so that we don't use an initial condition file that is incompatible with irrigation.

**Example 4-2. Example Irrigation Simulation**

```
> cd scripts
# Note here we do a CLMSP simulation as that is what has been validated
> ./create_newcase -case IRRIG -res f19_g16 -compset I -mach yellowstone_intel
> cd IRRIG
# Append "-irrig" to CLM_BLDNML_OPTS in env_run.xml (you could also use an editor)
> ./xmlchange CLM_BLDNML_OPTS="-irrig" -append
# Change to startup type so uses spunup initial conditions file for irrigation if it exists
# By default the model will do a hybrid startup with an initial condition file
# incompatible with the irrigation surface dataset.
> ./xmlchange RUN_TYPE=startup
> ./cesm_setup
# Now build and run normally
> ./IRRIG.build
> ./IRRIG.submit
```

**Caution**

We have only run the irrigation model with CLMSP (i.e. without the CN model). We recommend that if you want to run the irrigation model with CN, that you do a spinup as outlined in the examples below. But, more than that you may need to make the adjustments we discuss in the Section called *Build-Namelist options that should NOT be exercised:* in *What is scientifically validated and functional in CLM4 in CESM1.1.1?*.

# Spinning up the Satellite Phenology Model (CLMSP spinup)

To spin-up the CLMSP model you merely need to run CLMSP for 50 simulation years starting from arbitrary initial conditions. You then use the final restart file for initial conditions in other simulations. Because, this is a straight forward operation we will NOT give the details on how to do that here, but

leave it as an exercise for the reader. See the Example 4-5 as an example of doing this as the last step for CLMCN.

# Spinning up the biogeochemistry Carbon-Nitrogen Model (CN spinup)

To get the CLMCN model to a steady state, you first run it from arbitrary initial conditions using the "accelerated decomposition spinup" (-spinup AD in CLM **configure**) mode for 600 simulation years. After this you branch from this mode in the "exit spinup" (-spinup exit in CLM **configure**), run for a simulation year, and then save a restart from that and use it as initial conditions for further spinup of CN (at least 50 simulation years).

### Spinup of CLMCN

1. AD_SPINUP

   For the first step of running 600 years in "-spinup AD" mode, you will setup a case, and then edit the values in `env_build.xml` and `env_run.xml` so that the right configuration is turned on and the simulation is setup to run for the required length of simulation time. So do the following:

   **Example 4-3. Example AD_SPINUP Simulation**

```
> cd scripts
> ./create_newcase -case CN_spinup -res f19_g16 -compset ICN -mach yellowstone_intel
> cd CN_spinup
# Append "-spinup AD" to CLM_CONFIG_OPTS
> ./xmlchange CLM_CONFIG_OPTS="-spinup AD" -append
# The following sets CLM_FORCE_COLDSTART to "on" (you could also use an editor)
> ./xmlchange CLM_FORCE_COLDSTART=on
# Make the output history files only annual, by adding the following to the user_nl_clm namelist
> echo 'hist_nhtfrq = -8760' >> user_nl_clm
# Now setup
> ./cesm_setup -case
# Now build
> ./CN_spinup.build
# The following sets RESUBMIT to 30 times in env_run.xml (you could also use an editor)
# The following sets STOP_DATE,STOP_N and STOP_OPTION to Jan/1/601, 20, "nyears" in env_run.xml (you could also use an editor)
> ./xmlchange RESUBMIT=30,STOP_N=20,STOP_OPTION=nyears,STOP_DATE=6010101
# Now run normally
> ./CN_spinup.submit
```

   Afterwards save the last restart file from this simulation to use in the next step.

2. EXIT_SPINIP

   **Example 4-4. Example EXIT_SPINUP Simulation**

```
> cd scripts
> ./create_newcase -case CN_exitspinup -res f19_g16 -compset ICN -mach yellowstone_intel
> cd CN_exitspinup
# Append "-spinup exit" to CLM_CONFIG_OPTS
> ./xmlchange CLM_CONFIG_OPTS="-spinup exit" -append
# Change run type to branch and branch from the last year of the last simulation
> ./xmlchange RUN_TYPE=branch,RUN_REFCASE=CN_spinup,RUN_REFDATE=0601-01-01,GET_REFCASE=FALSE
> ./cesm_setup
# Go ahead and build, so that the run directory is created
> ./CN_exitspinup.build
# Now, Copy the last restart files from the earlier case into your run directory
```

```
> cp /ptmp/$LOGIN/archive/CN_spinup/rest/CN_spinup.*.r*.0601-01-01-00000* /ptmp/$LOGIN/CN_exitspinup
# And copy the rpointer files for datm and drv from the earlier case
> cp /ptmp/$LOGIN/archive/CN_spinup/rest/rpointer.atm /ptmp/$LOGIN/CN_exitspinup
> cp /ptmp/$LOGIN/archive/CN_spinup/rest/rpointer.drv /ptmp/$LOGIN/CN_exitspinup
# The following sets STOP_OPTION to "nyears" in env_run.xml (you could also use an editor)
> ./xmlchange STOP_OPTION=nyears,STOP_N=1
# Now run normally
> ./CN_exitspinup.submit
```

3. Final spinup

   Next save the last restart file from this step and use it as the "finidat" file to use for one more spinup for at least 50 years in normal mode. So do the following:

   **Example 4-5. Example Final CN Spinup Simulation**

```
> cd scripts
> ./create_newcase -case CN_finalspinup -res f19_g16 -compset ICN -mach yellowstone_intel
> cd CN_finalspinup
# The following sets CLM_FORCE_COLDSTART to "on" in env_build.xml (you could also use an editor)
> ./xmlchange CLM_FORCE_COLDSTART=on
# Now, Copy the last CLM restart file from the earlier case into your run directory
> cp /ptmp/$LOGIN/archive/CN_exitspinup/rest/CN_exitspinup.clm*.r*.0602-01-01-00000.nc \
/ptmp/$LOGIN/CN_finalspinup
# And copy the rpointer files for datm and drv from the earlier case
> cp /ptmp/$LOGIN/archive/CN_exitspinup/rest/rpointer.atm /ptmp/$LOGIN/CN_finalspinup
> cp /ptmp/$LOGIN/archive/CN_exitspinup/rest/rpointer.drv /ptmp/$LOGIN/CN_finalspinup
# Set the finidat file to the last restart file saved in previous step
> echo ' finidat = "CN_exitspinup.clm2.r.0602-01-01-00000.nc"' > user_nl_clm
# Now setup
> ./cesm_setup
> $EDITOR Buildconf/clm.buildnml.csh
> Now build
> .CN_finalspinup.build
# The following sets RESUBMIT to 5 times in env_run.xml (you could also use an editor)
# The following sets STOP_N and STOP_OPTION to 10 and "nyears" in env_run.xml (you could also use an editor)
> ./xmlchange RESUBMIT=5,STOP_OPTION=nyears,STOP_N=10
> Now run as normal
> .CN_finalspinup.submit
```

   To assess if the model is spunup plot trends of CN variables of interest. If you see a trend, you may need to run the simulation longer. Finally save the restart file from the end of this simulation to use as an "finidat" file for future simulations.

# Spinning up the Carbon-Nitrogen Dynamic Global Vegetation Model (CNDV spinup)

To spinup the CLM CNDV model -- you first follow the procedures above to spinup the CN model. Then you take the CN initial state file you created for the spinup with just CN, and run CNDV for 200 more years. We've provided such spunup files for two resolutions (f09 and f19) and two time-periods (1850 and 2000), so in this example we will use the files provided to start from. We've also provided a spinup file at f19 resolution for CNDV, hence the following is NOT required when running at f19. If you were to

start from your own CLMCN spunup files -- the procedure would require some modification. There are no compsets using CNDV, so in `env_build.xml` change CLM_CONFIG_OPTS to `-bgc cndv`.

**Example 4-6. Example CNDV Spinup Simulation**

```
> cd scripts
> ./create_newcase -case CNDV_spinup -res f09_g16 -compset ICN -mach yellowstone_intel
> cd CNDV_spinup
# Set run type to startup and do a cold start and CLM_CONFIG_OPTS to CNDV
> ./xmlchange RUN_TYPE=startup,CLM_CONFIG_OPTS="-bgc cndv"
# Make the default primary history file annual and add an annual 1D vector auxiliary file
# By putting the following in a user_nl_clm file.
> cat << EOF >> user_nl_clm
 hist_nhtfrq = -8760, -8760
 hist_mfilt  =     1, 1
 hist_fincl2 = 'TLAI', 'TSAI', 'HTOP', 'HBOT', 'NPP'
 hist_dov2xy = .true., .false.
> ./cesm_setup
# NOTE: If you were using your own CN spinup files you would edit the namelist to use it
# $EDITOR Buildconf/clm.buildnml.csh
#
# Now build and run as normal
> ./CNDV_spinup.build
# The following sets RESUBMIT to 10 times, STOP_N/STOP_OPTION to 20 years in env_run.xml (you could also use an editor)
> ./xmlchange RESUBMIT=10,STOP_OPTION=nyears,STOP_N=20
# Make sure you turn archiving on, so you save your files to long term archival
> ./xmlchange DOUT_L_MS=TRUE
> ./CNDV_spinup.submit
```

In a data analysis tool you should examine the auxiliary file and examine the `pfts1d_wtgcell` to see where and what types of vegetation have been established. See the caution in Example 1-7 for more information on visualizing and analyzing 1D vector fields.

**Note:** CNDV also writes out two vector fields to "hv" auxiliary files, on an annual basis by default.

**Note:** We've provided a spinup file for CNDV at f19 resolution, you could also use **interpinic** to interpolate this file to other resolutions.

# Running with MOAR data as atmospheric forcing to spinup the model

Because it takes so long to spinup the CN model (as we just saw previously), if you are doing fully coupled simulations with active atmosphere and ocean, you will want to do the spinup portion of this "offline". So instead of doing expensive fully coupled simulations for the spinup duration, you run CLM in a very cheap "I" compset using atmospheric forcing from a shorter fully coupled simulation (or a simulation run previously by someone else).

In this example we will use the `I1850SPINUPCN` compset to setup CLM to run with atmospheric forcing from a previous fully coupled simulation with data that is already stored on disk on yellowstone. There

are several simulations that have high frequency data for which we can do this. You can also do this on a machine other than yellowstone, but would need to download the data from the Earth System Grid and change the datapath similar to Example 4-9.

**Example 4-7. Example Simulation with MOAR Data on yellowstone**

```
> cd scripts
> ./create_newcase -case MOARforce1850 -res f19_g16 -compset I1850SPINUPCN -mach yellowstone_intel
> cd MOARforce1850
# The following sets the casename to point to for atm forcing (you could also use an editor)
> ./xmlchange DATM_CPL_CASE=b40.1850.track1.1deg.006a
# The following sets the align year and years to run over for atm forcing
#  (you could also use an editor)
> ./xmlchange DATM_CPL_YR_ALIGN=1,DATM_CPL_YR_START=960,DATM_CPL_YR_END=1030
> ./cesm_setup
# Now build and run as normal
> ./MOARforce1850.build
> ./MOARforce1850.submit
```

**Caution**

Because of bug 1339 (see the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file on this) you can't run with 83 or more years of forcing. If you do need to run with more years of forcing, you'll need to address the issue as outlined in the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file.

# Running with your own previous simulation as atmospheric forcing to spinup the model

Another way that you might want to spinup the model is to run your own simulation for a relatively short period (either a B, E, or F compset) and then use it as forcing for your "I" case later. By only running 20 to 50 years for the fully coupled case, you'll save a substantial amount of computer time rather than running the entire spinup period with a fully coupled model.

The first thing we need to do is to run a fully coupled case and save the atmospheric coupling fields on a three hourly basis. In this example, we will run on yellowstone and archive the data to a local disk that we can then use in the next simulation.

**Example 4-8. Example Fully Coupled Simulation to Create Data to Force Next Example Simulation**

```
> cd scripts
> ./create_newcase -case myBCN1850 -res f09_g16 -compset B1850CN  -mach yellowstone_intel
> cd myBCN1850
> ./cesm_setup
# Set histaux_a2x3hr to .true. in your user_nl_cpl output from the atmosphere model
# will be saved 3 hourly
echo "histaux_a2x3hr=.true." >> user_nl_cpl
# Now build
> ./myBCN1850.build
# The following sets the archival disk space (you could also use an editor)
> ./xmlchange DOUT_S_ROOT='/glade/home/$USER/$CASE'
```

```
# Make sure files are archived to disk, but NOT to long term storage
# (you could also use an editor)
> ./xmlchange DOUT_S=TRUE,DOUT_L_MS=FALSE
# Set the run length to run a total of 20 years (you could also use an editor)
> ./xmlchange RESUBMIT=9,STOP_OPTION=nyears,STOP_N=2
# Now run as normal
> ./myBCN1850.submit
```

**Caution**

Because of bug 1733 (see the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file on this) you'll need to edit the driver code in order for it to produce the correct list of fields needed to run the model later.

Now we run an I compset forced with the data from the previous simulation using the `CPLHIST3HrWx` option to DATM_MODE. See the Section called *`CPLHIST3HrWx` mode and it's DATM settings* in Chapter 1 for more information on the DATM settings for `CPLHIST3HrWx` mode.

**Example 4-9. Example Simulation Forced with Data from the Previous Simulation**

```
> cd scripts
> ./create_newcase -case frcwmyBCN1850 -res f09_g16 -compset I1850SPINUPCN -mach yellowstone_intel
> cd frcWmyBCN1850
# The following sets the casename to point to for atm forcing (you could also use an editor)
> ./xmlchange DATM_CPLHIST_CASE="myBCN1850"
# The following sets the align year and years to run over for atm forcing
#  (you could also use an editor)
> ./xmlchange DATM_CPLHIST_YR_ALIGN="1",DATM_CPLHIST_YR_START=1,DATM_CPLHIST_YR_END=20
# Set the strm_datdir in the namelist_defaults_datm.xml
# file to the archival path of the case above in the form of: /glade/home/achive/$USER/$DATM_CPLHIST_CASE/cpl/hist
# NOTE: THIS WILL CHANGE THE PATH FOR ALL I1850SPINUPCN COMPSET CASES MADE AFTER THIS!
> $EDITOR ../../models/atm/datm/bld/namelist_files/namelist_defaults_datm.xml
> ./cesm_setup
# Now build and run as normal
> ./frcwmyBCN1850.build
> ./frcwmyBCN1850.submit
```

**Note:** We did this by editing the "namelist_defaults_datm.xml" which will change the settings for ALL future I1850SPINUPCN cases you run. You could also do this by editing the path in the resulting streams text files in the CaseDocs directory, and then create a "user_" streams file with the correct path. This would change the streams file JUST for this case. The steps do it this way are:

```
> ./preview_namelists
> cp CaseDocs/datm.streams.txt.CPLHIST3HrWx.Precip            user_datm.streams.txt.CPLHIST3HrWx.Precip
> cp CaseDocs/datm.streams.txt.CPLHIST3HrWx.Solar             user_datm.streams.txt.CPLHIST3HrWx.Solar
> cp CaseDocs/datm.streams.txt.CPLHIST3HrWx.nonSolarNonPrecip user_datm.streams.txt.CPLHIST3HrWx.nonSolarNonPrecip
# Change the <fieldInfo> field <filePath> to point to the correct directory i.e.: /glade/home/achive/$USER/$DATM_CPLHIST_CASE/cpl/hist
> $EDITOR user_datm.streams.txt.CPLHIST3HrWx.*
> ./preview_namelists
# Then make sure the CaseDocs/datm.streams.txt.CPLHIST3HrWx.* files have the correct path
```

# Running stand-alone CLM with transient historical $CO_2$ concentration

In this case you want to run a simulation with stand-alone CLM responding to changes in $CO_2$ for a historical period. For this example, we will start with the "I_1850-2000_CN" compset that has transient: land-use, Nitrogen and Aerosol deposition already. You could also use another compset if you didn't want these other features to be transient. In order to get $CO_2$ to be transient we need to add a new streams file and add it to the list of streams in the user_nl_datm file. You also need a NetCDF datafile that datm can read that gives the variation. You could supply your own file, but we have a standard file that is used by CAM for this and our example will make use of this file.

**Note:** Most everything here has to do with changing datm rather than CLM to allow this to happen. As such the user that wishes to do this should first become more familiar with datm and read the CESM Data Model User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/data8) especially as it pertains to the datm.

## Warning

This section documents the process for doing something that is non-standard. There may be errors with the documentation and process, and you may have to do some work before all of this works for you. If that is the case, we recommend that you do further research into understanding the process and the files, as well as understanding the datm and how it works. You may have to read documentation found in the code for datm as well as "csm_share".

The datm has "streams" files that have rough XML-like syntax and specify the location and file to get data from, as well as information on the variable names and the data locations of the grid points. The datm expects specific variable names and the datm "maps" the expected variable names from the file to the names expected by datm. The file we are working with here is a file with a single-point, that covers the entire globe (so the vertices go from -90 to 90 degrees in latitude and 0 to 360 degrees in longitude). Since it's a single point it's a little easier to work with than datasets that may be at a given horizontal resolution. The datm also expects that variables will be in certain units, and only expects a limited number of variables so arbitrary fields can NOT be exchanged this way. However, the process would be similar for datasets that do contain more than one point.

The three things that are needed: a domain file, a data file, and a streams text file. The domain file is a CF-compliant NetCDF file that has information on the grid points (latitudes and longitudes for cell-centers and vertices, mask , fraction, and areas). The datafile is a CF-compliant NetCDF file with the data that will be mapped. The streams text file is the XML-like file that tells datm how to find the files and how to map the variables datm knows about to the variable names on the NetCDF files. Note, that in our case the domain file and the data file are the same file. In other cases, the domain file may be separate from the data file.

First we are going to create a case, and we will edit the `user_nl_datm` so that we add a $CO_2$ data stream in. There is a streams text file available in `models/lnd/clm/doc/UsersGuide/datm.streams.txt.co2`, that includes file with a $CO_2$ time-series from 1765 to 2007.

**Example 4-10. Example Transient Simulation with Historical $CO_2$**

```
> cd scripts
> ./create_newcase -case DATM_CO2_TSERIES -res f19_g16 -compset I_1850-2000_CN \
-mach yellowstone_intel
> cd DATM_CO2_TSERIES
# Set CCSM_BGC to CO2A so that CO2 will be passed from atmosphere to land
# Set CLM_CO2_TYPE to diagnostic so that the land will use the value sent from the atmosphere
> ./xmlchange CCSM_BGC=CO2A,CLM_CO2_TYPE=diagnostic
> ./cesm_setup
# Create the streams file for CO2
> cat << EOF >> datm.streams.txt.co2tseries
<streamstemplate>
      <general_comment>
      This is a streams file to pass historical CO2 from datm8 to the other
      surface models. It reads in a historical dataset derived from data used
      by CAM. The getco2_historical.ncl script in models/lnd/clm2/tools/ncl_scripts
      was used to convert the CAM file to a streams compatible format (adding domain
      information and making CO2 have latitude/longitude even if only for a single
      point.
      </general_comment>
<stream>
      <comment>
        Input stream description file for historical CO2 reconstruction data

        04 March 2010: Converted to form that can be used by datm8 by Erik Kluzek
        18 December 2009: Prepared by B. Eaton using data provided by
        Jean-Francois Lamarque. All variables except f11 are directly from
        PRE2005_MIDYR_CONC.DAT. Data from 1765 to 2007 with 2006/2007 just
        a repeat of 2005.
      </comment>
      <dataSource>
         CLMNCEP
      </dataSource>
      <domainInfo>
         <variableNames>
            time    time
            lonc    lon
            latc    lat
            area    area
            mask    mask
         </variableNames>
         <filePath>
            $CSMDATA/atm/datm7/CO2
         </filePath>
         <fileNames>
            fco2_datm_1765-2007_c100614.nc
         </fileNames>
      </domainInfo>
      <fieldInfo>
         <variableNames>
            CO2        co2diag
         </variableNames>
         <filePath>
            $CSMDATA/atm/datm7/CO2
         </filePath>
         <fileNames>
            fco2_datm_1765-2007_c100614.nc
         </fileNames>
      </fieldInfo>
</stream>
</streamstemplate>

EOF
# And copy it to the run directory
> cp datm.streams.txt.co2tseries $RUNDIR
# Run preview namelist so we have the namelist in CaseDocs
> ./preview_namelists
```

The first thing we will do is to edit the user_nl_datm file to add a CO2 file stream in. To do this we will copy a `user_nl_datm` in with the changes needed. The file `addco2_user_nl_datm.user_nl` is in `models/lnd/clm/doc/UsersGuide` and looks like this...

```
dtlimit = 1.5,1.5,1.5,1.5,1.5
fillalgo = 'nn','nn','nn','nn','nn'
fillmask = 'nomask','nomask','nomask','nomask','nomask'
mapalgo = 'bilinear','bilinear','bilinear','bilinear','nn'
mapmask = 'nomask','nomask','nomask','nomask',nomask'
streams = "datm.streams.txt.CLM_QIAN.Solar 1895 1948 1972  ", "datm.streams.txt.CLM_QIAN.Precip 1895 1948 1972  ",
        "datm.streams.txt.CLM_QIAN.TPQW 1895 1948 1972  ", "datm.streams.txt.presaero.trans_1850-2000 1849 1849 2006",
        "datm.streams.txt.co2 1766 1766 2005 "
taxmode = 'cycle','cycle','cycle','cycle','extend'
tintalgo = 'coszen','nearest','linear','linear','linear'
```

You just copy this into your case directory. But, also compare it to the version in `CaseDocs` to make sure the changes are just to add in the new $CO_2$ stream. Check to see that filenames, and start, end and align years are correct.

```
> cp ../../models/lnd/clm/doc/UsersGuide/addco2_user_nl_datm.user_nl user_nl_datm
> diff user_nl_datm CaseDocs/datm_atm_in
```

Once, you've done that you can build and run your case normally.

## Warning

This procedure assumes you are using a `I_1850-2000_CN` compset out of the box, with DATM_PRESAERO equal to `trans_1850-2000`. So it assumes standard Qian atmosphere forcing, and transient prescribed aerosols from streams files. If your case changes anything here your `user_nl_datm` file will need to be adjusted to work with it.

**Note:** The intent of the `user_nl_datm` is to adds an extra streams file for $CO_2$ to the end of the streams variable, and other arrays associated with streams (adding mapalgo as a new array with bilinear for everything, but the $CO_2$ file which should be "nn" for nearest neighbor). Other variables should be the same as the other stream values.

## Warning

The streams file above is hard-coded for the path of the file on NCAR computers. To use it on an outside machine you'll need to edit the filepath in the streams file to point to the location where you have the file.

After going through these steps, you will have a case where you have datm reading in an extra streams text file that points to a data file with $CO_2$ data on it that will send that data to the CLM.

# Chapter 5. How to run Single-Point/Regional cases

The CLM also allows you to set up and run cases with a single-point or a local region as well as global resolutions. This is often useful for running quick cases for testing, evaluating specific vegetation types, or land-units, or running with observed data for a specific site. There are three different ways to do this: normal-supported site, PTS_MODE, CLM_USRDAT_NAME, and with PTCLM.

*normal supported site* -- to run for a supported single point or regional dataset.
*PTS_MODE* -- to run for a single point using global datasets.
*CLM_USRDAT_NAME* -- to run using your own datasets (single-point or regional).

**Note:** PTS_MODE only works for a single point, while the other two options can also work for regional datasets as well.

## Which Single Point Option Should I choose?

Running for a *normal supported site* is a great solution, if one of the supported single-point/regional datasets, is your region of interest (see the Section called *Running Supported Single-point/Regional Datasets*). All the datasets are created for you, and you can easily select one and run, out of the box with it using a supported resolution from the top level of the CESM scripts. The problem is that there is a very limited set of supported datasets. You can also use this method for your own datasets, but you have to create the datasets, and add them to the XML database in scripts, CLM and to the DATM. This is worthwhile if you want to repeat many multiple cases for a given point or region.

In general the Section called *Running PTS_MODE configurations* is the quick and dirty method that gets you started without having to create datasets -- but has limitations. It's good for an initial attempt at seeing results for a point of interest, but since you can NOT restart with it, it's usage is limited. It is the quickest method as you can create a case for it directly from **create_newcase**. Although you can't restart, running a single point is very fast, and you can run for long simulation times even without restarts. If you need restarts a good solution is to use **getregional_datasets.pl** and *CLM_USRDAT_NAME* which can get you running almost as quickly as well as *PTS_MODE*. Like *PTS_MODE* the Section called *Creating a set of regional datasets from existing global datasets* in Chapter 2 only runs for points that exist within a global dataset.

Next, *CLM_USRDAT_NAME* is the best way to setup cases quickly where you have to create your own datasets (see the Section called *Creating your own single-point/regional surface datasets*). With this method you don't have to change DATM or add files to the XML database -- but you have to follow a strict naming convention for files. However, once the files are named and in the proper location, you can easily setup new cases that use these datasets. This is good for treating all the required datasets as a "group" and for a particular model version. For advanced CLM developers who need to track dataset changes with different model versions you would be best off adding these datasets as supported datasets with the "normal supported datasets" method.

Finally, if you also have meteorology data that you want to force your CLM simulations with you'll need to setup cases as described in the Section called *Running with your own atmosphere forcing*. You'll need to create CLM datasets either according to *CLM_USRDAT_NAME*. You may also need to modify DATM to use your forcing data. And you'll need to change your forcing data to be in a format that DATM can use.

# Running PTS_MODE configurations

PTS_MODE enables you to run the model using global datasets, but just picking a single point from those datasets and operating on it. It can be a very quick way to do fast simulations and get a quick turnaround.

To setup a PTS_MODE simulation you use the "-pts_lat" and "-pts_lon" arguments to **create_newcase** to give the latitude and longitude of the point you want to simulate for (the code will pick the point on the global grid nearest to the point you give. Here's an example to setup a simulation for the nearest point at 2-degree resolution to Boulder Colorado.

```
> cd scripts
> ./create_newcase -case testPTS_MODE -res f19_g16 -compset I -mach yellowstone_intel \
-pts_lat 40.0 -pts_lon -105
> cd testPTS_MODE
# We make sure the model will start up cold rather than using initial conditions
> ./xmlchange CLM_FORCE_COLDSTART=on,RUN_TYPE=startup
```

Then setup, build and run as normal. We make sure initial conditions are NOT used since PTS_MODE currently CAN NOT run with initial conditions.

**Important:** By default it sets up to run with MPILIB=mpi-serial (in the `env_build.xml` file) turned on, which allows you to run the model interactively. On some machines this mode is NOT supported and you may need to change it to FALSE before you are able to build.

**Warning**

PTS_MODE currently does *NOT* restart nor is it able to startup from global initial condition files. See bugs "1017 and 1025" in the `models/lnd/clm/doc/KnownLimitationss` (../KnownLimitations) file.

**Note:** You can change the point you are simulating for at run-time by changing the values of PTS_LAT and PTS_LON in the `env_run.xml` file.

**Note:** Note, that when running with PTS_MODE the number of processors is automatically set to one. When running a single grid point you can only use a single processor. You might also want to set the `env_build.xml` variable: MPILIB=mpi-serial to `TRUE` so that you can also run interactively without having to use MPI to start up your job.

# Warning about Running with a Single-Processor on a Batch Machine

This problem always comes up when running for a single point, because you can only use a single-processor, but may come up in other instances when you are running with one processor. This applies to all the different ways of running in single-point mode.

> **Note:** On many machines is that some batch ques have a minimum number of nodes or processors that can be used. On these machines you may have to change the queue and possibly the time-limits of the job, to get it to run in the batch que. On yellowstone this is done for you automatically, and the "caldera" queue is used for such single-processor simulations.

Another way to get around this problem is to run the job interactively using MPILIB=mpi-serial so that you don't submit the job to the batch que. For single point mode you also may want to consider using a smaller workstation or cluster, rather than a super-computer, because you can't take advantage of the multi-processing power of the super-computer anyway.

# Running Supported Single-point/Regional Datasets

In addition to PTS_MODE the CLM supports running using single-point or regional datasets that are customized to a particular region. In the section below we tell the user how to create their own dataset, but we also support a small number of single-point and regional datasets that are ready to setup and run in the CESM modeling system.

To get the list of supported dataset resolutions do this:

```
> cd models/lnd/clm/doc
> ../bld/build-namelist -res list
```

Which results in the following:

```
build-namelist - valid values for res (Horizontal resolutions
Note: 0.1x0.1, 0.5x0.5, 5x5min, 10x10min, 3x3min and 0.33x0.33 are only used for CLM tools):
                Values: default 512x1024 128x256 64x128 48x96 32x64 8x16 94x192 0.23x0.31  \
   0.47x0.63 0.9x1.25 1.9x2.5 2.5x3.33 4x5 10x15 5x5_amazon 1x1_tropicAtl 1x1_camdenNJ  \
   1x1_vancouverCAN 1x1_mexicocityMEX 1x1_asphaltjungleNJ 1x1_brazil 1x1_urbanc_alpha 1x1_numaIA  \
   1x1_smallvilleIA 0.1x0.1 0.5x0.5 3x3min 5x5min 10x10min 0.33x0.33 ne4np4 ne16np4 ne30np4 ne60np4  \
   ne120np4 ne240np4
                Default = 1.9x2.5
                (NOTE: resolution and mask and other settings may influence what the default is)
```

The resolution names that have an underscore in them ("_") are all single-point or regional resolutions.

To run for the Brazil test site do the following:

**Example 5-1. Example of running CLM over a single-point test site in Brazil with the default Qian atmosphere data forcing.**

```
> cd scripts
> set SITE=1x1_brazil
> ./create_newcase -case testSPDATASET -res $SITE -compset I \
-mach yellowstone_intel
> cd testSPDATASET
```

Then setup, build and run normally.

Then to run for the urban Mexico City Mexico test site that also has atmosphere forcing data, but to run it with the Qian forcing data, but over the period for which it's own forcing data is provided do the following:

**Example 5-2. Example of running CLM over the single-point of Mexicocity Mexico with the default Qian atmosphere data forcing.**

```
> cd scripts
# Set a variable to the site you want to use (as it's used several times below)
> set SITE=1x1_mexicocityMEX
> ./create_newcase -case testSPDATASET -res $SITE -compset I \
-mach yellowstone_intel
> cd testSPDATASET
# Set DATM prescribed aerosols to single-point dataset
# Will then use the dataset with just the point for this $SITE
> ./xmlchange DATM_PRESAERO=pt1_pt1
```

Then setup, build and run normally.

**Important:** Just like PTS_MODE above, By default it sets up to run with MPILIB=mpi-serial (in the `env_build.xml` file) turned on, which allows you to run the model interactively. On some machines this mode is NOT supported and you may need to change it to FALSE before you are able to build.

## Warning

See the Section called *Warning about Running with a Single-Processor on a Batch Machine* for a warning about running single-point jobs on batch machines.

**Note:** Note, that when running a `pt1_pt1` resolution the number of processors is automatically set to one. When running a single grid point you can only use a single processor. You might also want to set the `env_build.xml` variable: MPILIB=mpi-serial to `TRUE` so that you can also run interactively without having to use mpi to start up your job.

## Running Supported Single-point Datasets that have their own Atmospheric Forcing

Of the supported single-point datasets we have three that also have atmospheric forcing data that go with them: Mexico City (Mexico), Vancouver, (Canada, British Columbia), and urbanc_alpha (test data for an Urban inter-comparison project). Mexico city and Vancouver also have "#ifdef" in the source code for them to work with modified urban data parameters that are particular to these locations. They can be turned on by using the CLM_CONFIG_OPTS `env_build.xml` variable to set the "-sitespf_pt" option in the CLM **configure**. To turn on the atmospheric forcing for these datasets, you set the `env_run.xml` DATM_MODE variable to `CLM1PT`, and then the atmospheric forcing datasets will be used for the point picked.

When running with datasets that have their own atmospheric forcing you need to be careful to run over the period that data is available. If you have at least one year of forcing it will cycle over the available data over and over again no matter how long of a simulation you run. However, if you have less than a years worth of data (or if the start date doesn't start at the beginning of the year, or the end date doesn't end at the end of the year) then you won't be able to run over anything but the data extent. In this case you will need to carefully set the RUN_STARTDATE, START_TOD and STOP_N/STOP_OPTION variables for your case to run over the entire time extent of your data. For the supported data points, these values are in the XML database and you can use the **queryDefaultNamelist.pl** script to query the values and set them for your case (they are set for the three urban test cases: Mexicocity, Vancouver, and urbanc_alpha).

In the example below we will show how to do this for the Vancouver, Canada point.

**Example 5-3. Example of running CLM over the single-point of Vancouver Canada with supplied atmospheric forcing data for Vancouver.**

```
> cd scripts
# Set a variable to the site you want to use (as it's used several times below)
> set SITE=1x1_vancouverCAN
# Create a case at the single-point resolutions with their forcing
> ./create_newcase -case testSPDATASETnAtmForcing -res $SITE -compset I1PT \
-mach yellowstone_intel
> cd testSPDATASETnAtmForcing
# Set namelist options for urban test site
> ./xmlchange CLM_NML_USE_CASE=stdurbpt_pd
# Figure out the start and end date for this dataset
# You can do this by examining the datafile.
> set STOP_N=330
> set START_YEAR=1992
> set STARTDATE=${START_YEAR}-08-12
> @ NDAYS = $STOP_N / 24
> ./xmlchange RUN_STARTDATE=$STARTDATE,STOP_N=$STOP_N,STOP_OPTION=nsteps
# Set the User namelist to set the output frequencies of the history files
# Setting the stdurbpt use-case option create three history file streams
# The frequencies and number of time-samples needs to be set
> cat << EOF > user_nl_clm
 hist_mfilt = $NDAYS,$STOP_N,$STOP_N
 hist_nhtfrq = -1,1,1
EOF
# Set DATM prescribed aerosols to single-point dataset
# Will then use the dataset with just the point for this site
> ./xmlchange DATM_PRESAERO=pt1_pt1
> ./cesm_setup
```

### Caution

If you don't set the start-year and run-length carefully as shown above the model will abort with a "dtlimit error" in the atmosphere model (see bug 1110 in the `models/lnd/clm/doc/KnownLimitationss` (../KnownLimitations) file for documentation on this). Since, the forcing data for this site (and the MexicoCity site) is less than a year, the model won't be able to run for a full year. The `1x1_urbanc_alpha` site has data for more than a full year, but neither year is complete hence, it has the same problem (see the problem for this site above).

**Important:** Just like PTS_MODE above, By default it sets up to run with MPILIB=mpi-serial (in the `env_build.xml` file) turned on, which allows you to run the model interactively. On some machines this mode is NOT supported and you may need to change it to FALSE before you are able to build.

### Warning

See the Section called *Warning about Running with a Single-Processor on a Batch Machine* for a warning about running single-point jobs on batch machines.

**Note:** Note, that when running a `pt1_pt1` resolution the number of processors is automatically set to one. When running a single grid point you can only use a single processor. You might also want to set the `env_build.xml` variable: MPILIB=mpi-serial to `TRUE` so that you can also run interactively without having to use mpi to start up your job.

# Creating your own single-point/regional surface datasets

The file: Quickstart.userdatasets (../Quickstart.userdatasets) in the `models/lnd/clm/doc` directory gives guidelines on how to create and run with your own single-point or regional datasets. Below we reprint the above guide.

```
          Quick-Start to using your own datasets in clm4
          ==============================================

Assumptions: You are already familiar with the use of the cpl7 scripts
             for creating cases to run with "standalone" clm. See the
             Quickstart.GUIDE and the README files and documentation in
             the scripts directory for more information on this process.
             We also assume that the env variable $CSMDATA points to the
             location of the standard datasets for your machine
             (/fis/cgd/cseg/csm/inputdata on bluefire). We also assume that the
             following variables are used to point to the appropriate
             values that you want to use for your case. Mask is included
             as part of your resolution for your case, and SIM_YEAR and
             SIM_YEAR_RANGE will be set appropriately for the particular use
             case that you choose for your compset (i.e. 1850_control,
             20thC_transient etc.).

                 SIM_YEAR -------- Simulation year       (i.e. 1850, or 2000)
                 SIM_YEAR_RANGE -- Simulation year range (i.e. constant, or 1850-2000)
```

```
                MASK ------------ Land mask             (i.e. navy, USGS, or gx1v6)

Process:

       0.) Why do this?

       An alternative to the steps below, is to create your case, and hand-edit the
       relevant namelists as appropriate with your own datasets. One reason for
       the process below is so that we can do automated testing on dataset inclusion.
       But, it also provides the following functionality to the user:
           a.) New cases with the same datasets only require a small change to
               env_conf.xml and env_run.xml (steps 5,6, and 8)
           b.) You can clone new cases based on a working case, without having to
               hand-edit all of the namelists for the new case in the same way.
           c.) The process will check for the existence of files when cases are
               configured so you can have the scripts check that datasets exist
               rather than finding out at run-time after submitted to batch.
           d.) The process checks for valid namelists, and makes it less likely
               for you to put an error or typo in the namelists.
           e.) The *.input_data_list files will be accurate for your case,
               you can use the check_input_data script to do queries on the files.
           f.) Your dataset names will be closer to standard names, and easier
               for inclusion in standard clm (with the exception of creation dates).
           g.) The regional extraction script (see 3.b below) will automatically create
               files with names following this convention.

       1.) Create your own dataset area -- link it to standard dataset location

       Create a directory to put your own datasets (such as /ptmp/$USER/my_inputdata).
       Use the script link_dirtree to link the standard datasets into this location.
       If you already have complete control over the datasets in $CSMDATA -- you
       can skip this step.

         setenv MYCSMDATA /ptmp/$USER/my_inputdata
         scripts/link_dirtree $CSMDATA $MYCSMDATA

       If you do this you can find the files you've added with...

           find $MYCSMDATA -type f -print

       and you can find the files that are linked to the standard location with...

           find $MYCSMDATA -type l -print

       2.) Establish a "user dataset identifier name" string

       You need a unique identifier for your datasets for a given resolution,
       mask, area, simulation-year, and simulation year-range. The identifier
       can be any string you want -- but we have the following suggestions:

       Suggestions for global grids:

         setenv MYDATAID ${degLat}x${degLon}

       Suggestions for regional grids: either give the number of points in the grid

         setenv MYDATAID nxmpt_citySTATE
         setenv MYDATAID nxmpt_cityCOUNTRY
         setenv MYDATAID nxmpt_regionCOUNTRY
         setenv MYDATAID nxmpt_region

                       or give the total size of the gridcells

         setenv MYDATAID nxmdeg_citySTATE
         setenv MYDATAID nxmdeg_cityCOUNTRY

        for example: setenv MYDATAID 10x15 -- global 10x15 grid
                     setenv MYDATAID 1x1pt_boulderCO -- single-point for Boulder CO
                     setenv MYDATAID 5x5pt_boulderCO -- 5x5 region around Boulder CO
                     setenv MYDATAID 1x1deg_boulderCO - 1x1 degree region around Boulder CO
                     setenv MYDATAID 13x12pt_f19_alaskaUSA1 - 13x12 gridcells from f19
                                                      (1.9x2.5) global resolution over Alaska

       3.) Add your own datasets in the standard locations in that area

       3.a) Create datasets using the standard tools valid for any specific points

       Use the tools in models/lnd/clm/tools to create new datasets. Tools
```

```
such as: mkgriddata, mksurfdata, mkdatadomain, and the regridding tools
in ncl_scripts

(see the models/lnd/clm/bld/namelist_files/namelist_defaults_usr_files.xml
 for the exact syntax for all files).

surfdata:    copy files into:
      $MYCSMDATA/lnd/clm2/surfdata/surfdata_${MYDATAID}_simyr${SIM_YEAR}.nc
domainfile:  copy files into:
      $MYCSMDATA/atm/datm7/domain.clm/domain.lnd.${MYDATAID}_${MASK}.nc

3.b) Use the regional extraction script to get regional datasets from the global ones
Use the getregional_datasets.pl script to extract out regional datasets of interest.
Note, the script works on all files other than the "finidat" file as it's a 1D vector file.

For example, Run the extraction for data from 52-73 North latitude, 190-220 longitude
that creates 13x12 gridcell region from the f19 (1.9x2.5) global resolution over Alaska.

         cd models/lnd/clm/tools/ncl_scripts
         ./getregional_datasets.pl -sw 52,190 -ne 73,220 -id $MYDATAID \
         -mycsmdata $MYCSMDATA

Repeat this process if you need files for multiple sim_year, and sim_year_range values.

4.) Setup your case

Follow the standard steps for executing "scripts/create_newcase" and customize
your case as appropriate.

i.e.

  ./create_newcase -case my_userdataset_test -res pt1_pt1 -compset I1850 \
         -mach bluefire

The above example implies that: MASK=gx1v6, SIM_YEAR=1850, and SIM_YEAR_RANGE=constant.
5.) Edit the env_run.xml in the case to point to your new dataset area

Edit DIN_LOC_ROOT in env_run.xml to point to $MYCSMDATA

  ./xmlchange DIN_LOC_ROOT=$MYCSMDATA

6.) Edit the env_conf.xml in the case to point to your user dataset identifier
name.

Edit CLM_USRDAT_NAME  to point to $MYDATAID

  ./xmlchange CLM_USRDAT_NAME=$MYDATAID
  ./xmlchange CLM_PT1_NAME=$MYDATAID

7.) Setup the case as normal

  ./setup

8.) Run your case as normal
```

## Using getregional_datasets.pl to get a complete suite of single-point/regional surface datasets from global ones

Use the regional extraction script to get regional datasets from the global ones The **getregional_datasets.pl** script to extract out regional datasets of interest. Note, the script works on all files other than the "finidat" file as it's a 1D vector file. The script will extract out a block of gridpoints from all the input global datasets, and create the full suite of input datasets to run over that block. The input datasets will be named according to the input "id" you give them and the id can then be used as input to CLM_USRDAT_NAME to create a case that uses it. See the section on CLM Script Configuration Items for more information on setting CLM_USRDAT_NAME (in Chapter 1). The list of

files extracted by their name used in the namelists are: `fsurdat`, `fpftdyn`, `stream_fldfilename_ndep`, and the DATM files `domainfile`, and `faerdep`. For more information on these files see the Table on required files.

The alternatives to using this script are to use PTS_MODE, discussed earlier, to use PTCLM discussed in the next chapter, or creating the files individually using the different file creation tools (given in the Tools Chapter). Creating all the files individually takes quite a bit of effort and time. PTS_MODE has some limitations as discussed earlier, but also as it uses global files, is a bit slower when running simulations than using files that just have the set of points you want to run over. Another advantage is that once you've created the files using this script you can customize them if you have data on this specific location that you can replace with what's already in these files.

The script requires the use of both "Perl" and "NCL". See the NCL Script section in the Tools Chapter on getting and using NCL and NCL scripts. The main script to use is a Perl script which will then in turn call the NCL script that actually creates the output files. The ncl script gets it's settings from environment variables set by the perl script. To get help with the script use "-help" as follows:

```
> cd models/lnd/clm/tools/ncl_scripts
> ./getregional_datasets.pl -help
```

The output of the above is:

```
SYNOPSIS
     getregional_datasets.pl [options]    Extracts out files for a single box region from the  \
   global
                                  grid for the region of interest. Choose a box determined by
                                  the NorthEast and SouthWest corners.
REQUIRED OPTIONS
     -mydataid "name" [or -id]    Your name for the region that will be extracted.               \
   (REQUIRED)
                                  Recommended name: grid-size_global-resolution_location  \
   (?x?pt_f??_????)
                                  (i.e. 12x13pt_f19_alaskaUSA for 12x13 grid cells from the f19  \
   global resolution over Alaska)
     -NE_corner "lat,lon" [or -ne] North East corner latitude and longitude                      \
   (REQUIRED)
     -SW_corner "lat,lon" [or -sw] South West corner latitude and longitude                      \
   (REQUIRED)
OPTIONS
     -debug [or -d]               Just debug by printing out what the script would do.
                                  This can be useful to find the size of the output area.
     -help [or -h]                Print usage to STDOUT.
     -mask "landmask"             Type of land-mask (i.e. navy, gx3v7, gx1v6 etc.) (default gx1v6)
 \
        -mycsmdata "dir"             Root directory of where to put your csmdata.
                                  (default /home/erik/inputdata or value of CSMDATA env variable)
     -nomv                        Do NOT move datasets to final location, just leave them in  \
   current directory
     -res "resolution"            Global horizontal resolution to extract data from (default  \
   1.9x2.5).
     -rcp "pathway"               Representative concentration pathway for future scenarios
                                  Only used when simulation year range ends in a future
                                  year, such as 2100.
                                  (default -999.9).
     -sim_year   "year"           Year to simulate for input datasets (i.e. 1850, 2000) (default  \
   2000)
     -sim_yr_rng "year-range"     Range of years for transient simulations
                                  (i.e. 1850-2000, 1850-2100,  or constant) (default constant)

     -verbose [or -v]             Make output more verbose.
```

The *required* options are: `-id`, `-ne`, and `-se`, for the output identifier name to use in the filenames, latitude and longitude of the Northeast corner, and latitude and longitude of the SouthEast corner (in degrees). Options that specify which files will be used are: `-mask`, `-res`, `-rcp`, `-sim_year`, and `-sim_yr_rng` for the land-mask to use, global resolution name, representative concentration pathway for future scenarios, simulation year, and simulation year range. The location of the input and output files will be determined by the option `-mycsmdata` (can also be set by using the environment variable `$CSMDATA`). If you are running on a machine like at NCAR where you do NOT have write permission to the CESM inputdata files, you should use the `scripts/link_dirtree` script to create soft-links of the original files to a location that you can write to. This way you can use both your new files you created as well as the original files and use them from the same location.

The remaining options to the script are `-debug`, and `-verbose`. `-debug` is used to show what would happen if the script was run, without creating the actual files. `-verbose` adds extra log output while creating the files so you can more easily see what the script is doing.

For example, Run the extraction for data from 52-73 North latitude, 190-220 longitude that creates 13x12 gridcell region from the f19 (1.9x2.5) global resolution over Alaska.

**Example 5-4. Example of running getregional_datasets.pl to get datasets for a specific region over Alaska**

```
> cd scripts
# First make sure you have a inputdata location that you can write to
# You only need to do this step once, so you won't need to do this in the future
> setenv MYCSMDATA $HOME/inputdata         # Set env var for the directory for input data
> ./link_dirtree $CSMDATA $MYCSMDATA
> cd ../models/lnd/clm/tools/ncl_scripts
> ./getregional_datasets.pl -sw 52,190 -ne 73,220 -id 13x12pt_f19_alaskaUSA -mycsmdata $MYCSMDATA
```

Repeat this process if you need files for multiple sim_year, resolutions, land-masks, and sim_year_range values.

**Warning**

See the Section called *Warning about Running with a Single-Processor on a Batch Machine* for a warning about running single-point jobs on batch machines.

**Note:** See the Section called *Managing Your Own Data-files* in Chapter 3 for notes about managing your data when using **link_dirtree**.

Now to run a simulation with the datasets created above, you create a single-point case, and set CLM_USRDAT_NAME to the identifier used above. Note that in the example below we set the number of processors to use to one (-pecount 1). For a single point, you should only use a single processor, but for a regional grid, such as the example below you could use up to the number of grid points (12x13=156 processors.

**Example 5-5. Example of using CLM_USRDAT_NAME to run a simulation using user datasets for a specific region over Alaska**

```
> cd scripts
# Create the case and set it to only use one processor
> ./create_newcase -case my_userdataset_test -res pt1_pt1 -compset I1850 \
-mach yellowstone_intel
> cd my_userdataset_test/
> setenv MYID=13x12pt_f19_alaskaUSA
> ./xmlchange DIN_LOC_ROOT_CSMDATA=$MYCSMDATA,CLM_USRDAT_NAME=$MYID,CLM_BLDNML_OPTS='-mask gx1v6'
> ./cesm_setup
```

# Running with your own atmosphere forcing

Here we want to run with our own customized datasets for CLM as well as running with our own supplied atmosphere forcing datasets. Thus we effectively combine the information from the Section called *Running Supported Single-point Datasets that have their own Atmospheric Forcing* with the Section called *Creating your own single-point/regional surface datasets*. First we need to follow the procedures in the Section called *Running Supported Single-point Datasets that have their own Atmospheric Forcing* to come up with CLM datasets that are customized for our point or region in question. This includes running **link_dirtree** to create a directory location where you can add your own files to it. Next, set DATM_MODE to `CLM1PT` and CLM_USRDAT_NAME to the id of the data you created. To see a list of what the filenames need to be see the section on setting CLM_USRDAT_NAME.

Next we need to setup the atmosphere forcing data in NetCDF format that can be read by DATM. There is a list of eight variables that are expected to be on the input files with the names and units on the following table (in the table TDEW and SHUM are optional fields that can be used in place of RH). In the table we also list which of the fields are required and if not required what the code will do to replace them. If the names of the fields are different or the list is changed from the standard list of eight fields: FLDS, FSDS, PRECTmms, PSRF, RH, TBOT, WIND, and ZBOT, the resulting streams file will need to be modified to take this into account (see an example streams file for this in Example 5-7 below).

**Table 5-1. Atmosphere Forcing Fields**

| Short-name | Description | Units | Required? | If NOT require how replaced |
|---|---|---|---|---|
| FLDS | incident longwave (FLDS) | W/m2 | No | calculates based Temperature, Pressure and Humidity |
| FSDS | incident solar (FSDS) | W/m2 | Yes | - |
| FSDSdif | incident solar (FSDS) diffuse | W/m2 | No | based on FSDS |
| FSDSdir | incident solar (FSDS) direct | W/m2 | No | based on FSDS |
| PRECTmms | precipitation (PRECTmms) | mm/s | Yes | - |

| Short-name | Description | Units | Required? | If NOT require how replaced |
|---|---|---|---|---|
| PSRF | pressure at the lowest atm level (PSRF) | Pa | No | assumes standard-pressur |
| RH | relative humidity at the lowest atm level (RH) | % | No | can be replaced with SHUM or TDEW |
| SHUM | specific humidity at the lowest atm level | kg/kg | Optional in place of RH | can be replaced with RH or TDE |
| TBOT | temperature at the lowest atm level (TBOT) | K (or can be C) | Yes | - |
| TDEW | dew point temperature | K (or can be C) | Optional in place of RH | can be replaced with RH or SHU |
| WIND | wind at the lowest atm level (WIND) | m/s | Yes | - |
| ZBOT | observational height | m | No | assumes 30 mete |

All of the variables should be dimensioned: time, lat, lon, with time being the unlimited dimension. The coordinate variable "time" is also required with CF-compliant units in days, hours, minutes, or seconds. It can also have a calendar attribute that can be "noleap" or "gregorian". Normally the files will be placed in the: `$MYCSMDATA/atm/datm7/CLM1PT_data/$MYUSRDAT` directory with separate files per month called `YYYY-MM.nc` where YYYY-MM corresponds to the four digit year and two digit month with a dash in-between. You also need a domain file that gives the coordinate information for the data that should be placed in: `$MYCSMDATA/atm/datm7/domain.lnd.$MYUSRDAT_USGS.nc`.

**Example 5-6. Example of setting up a case with your own atmosphere forcing**

```
> cd scripts
# First make sure you have a inputdata location that you can write to
# You only need to do this step once, so you won't need to do this in the future
> setenv MYCSMDATA $HOME/inputdata     # Set env var for the directory for input data
> ./link_dirtree $CSMDATA $MYCSMDATA
# Next create and move all your datasets into $MYCSMDATA with id $MYUSRDAT
# See above for naming conventions

#  Now create a single-point case
> ./create_newcase -case my_atmforc_test -res pt1_pt1 -compset I1850 \
-mach yellowstone_intel
> cd my_atmforc_test
# Set the data root to your inputdata directory, and set CLM_USRDAT_NAME
# to the user id you created for your datasets above
> ./xmlchange DIN_LOC_ROOT_CSMDATA=$MYCSMDATA,CLM_USRDAT_NAME=$MYUSRDAT
# Set the land-mask to USGS, so both clm and DATM can find files
> ./xmlchange CLM_BLDNML_OPTS='-mask USGS'
# Then set DATM_MODE to single-point mode so DATM will use your forcing datasets
# Put your forcing datasets into $MYCSMDATA/atm/datm7/CLM1PT_data/$MYUSRDAT
> ./xmlchange DATM_MODE=CLM1PT
> ./cesm_setup
# If the list of fields, or filenames, filepaths, or fieldnames are different
# you'll need to edit the DATM namelist streams file to make it consistent
```

```
> $EDITOR Buildconf/datm.buildnml.csh
```

**Warning**

See the Section called *Warning about Running with a Single-Processor on a Batch Machine* for a warning about running single-point jobs on batch machines.

**Note:** See the Section called *Managing Your Own Data-files* in Chapter 3 for notes about managing your data when using **link_dirtree**.

Now, we'll show an example of what the DATM streams file might look like for a case with your own forcing data with 3-hourly forcing. In this example, we'll leave off the fields: ZBOT, and FLDS so they'll be calculated as given in the Table 5-1 table above. We'll also include: FSDSdif and FSDSdir which aren't required, and we'll use TDEW in place of RH. In this example the datafiles are in NetCDF format and contain the fields: TA, Tdew, WS, PREC, Rg, Rgdir, Rgdif, and PRESS which are translated into the DATM internal names in this streams file. There is also a domain file that has the position information for this location. The normal assumption for `CLM1PT` mode in the DATM is that data is hourly or half-hourly and as such is often enough that using the data on the nearest time-stamp is reasonable and as such the data is in a single streams file (see the Section called *`CLM1PT` mode and it's DATM settings* in Chapter 1 for more information on the default settings for DATM and how to change them. If the data is less often three to six hours -- see Example 5-7 below, where you will need to modify the time-interpolation method as well as the time stamp offsets. In the example below we also have to divide the single stream file into three files to manage the time-stamps and time interpolation algorithm for the different types of data differently.

**Example 5-7. Example of DATM streams files with your own forcing for 3-hourly data**

Precipitation streams file (`clm1PT.1x1pt_lapazMEX.precip.stream.txt` file) .

```
<streamstemplate>
<stream>
      <dataSource>
         CLMNCEP
      </dataSource>
      <domainInfo>
         <variableNames>
            time    time
            xc      lon
            yc      lat
            area    area
            mask    mask
         </variableNames>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/domain.clm
         </filePath>
         <fileNames>
            domain.lnd.1x1pt_lapazMEX_navy.nc
         </fileNames>
      </domainInfo>
      <fieldInfo>
         <variableNames>
            PRECTmms PREC
         </variableNames>
```

```
         <offset>
            -5400
         </offset>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/CLM1PT_data/1x1pt_lapazMEX
         </filePath>
         <fileNames>
            2004-01.nc
            2004-02.nc
            2004-03.nc
.
.
.
            2009-12.nc
         </fileNames>
      </fieldInfo>
</stream>
</streamstemplate>
```

Solar streams file (`clm1PT.1x1pt_lapazMEX.solar.stream.txt` file).

```
<streamstemplate>
<stream>
      <dataSource>
         CLMNCEP
      </dataSource>
      <domainInfo>
         <variableNames>
            time    time
            xc      lon
            yc      lat
            area    area
            mask    mask
         </variableNames>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/domain.clm
         </filePath>
         <fileNames>
            domain.lnd.1x1pt_lapazMEX_navy.nc
         </fileNames>
      </domainInfo>
      <fieldInfo>
         <variableNames>
            FSDS     Rg
            FSDSdir  Rgdir
            FSDSdif  Rgdif
         </variableNames>
         <offset>
            -10800
         </offset>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/CLM1PT_data/1x1pt_lapazMEX
         </filePath>
         <fileNames>
            2004-01.nc
            2004-02.nc
            2004-03.nc
.
.
.
            2009-12.nc
         </fileNames>
      </fieldInfo>
</stream>
</streamstemplate>
```

Other fields streams file. (`clm1PT.1x1pt_lapazMEX.other.stream.txt` file) .

```
<streamstemplate>
<stream>
      <dataSource>
         CLMNCEP
      </dataSource>
      <domainInfo>
         <variableNames>
```

```
            time    time
            xc      lon
            yc      lat
            area    area
            mask    mask
         </variableNames>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/domain.clm
         </filePath>
         <fileNames>
            domain.lnd.1x1pt_lapazMEX_navy.nc
         </fileNames>
      </domainInfo>
      <fieldInfo>
         <variableNames>
            TBOT     TA
            TDEW     Tdew
            WIND     WS
            PSRF     PRESS
         </variableNames>
         <offset>
            -5400
         </offset>
         <filePath>
            $DIN_LOC_ROOT/atm/datm7/CLM1PT_data/1x1pt_lapazMEX
         </filePath>
         <fileNames>
            2004-01.nc
            2004-02.nc
            2004-03.nc
.
.
.
            2009-12.nc
         </fileNames>
      </fieldInfo>
</stream>
</streamstemplate>
```

Example streams namelist for the above streams files:

```
 &shr_strdata_nml
   dataMode       = 'CLMNCEP'
   domainFile     = '$DOMAINFILE'
   streams        = 'clm1PT.1x1pt_lapazMEX.solar.stream.txt  1 2004 2009 ',
                    'clm1PT.1x1pt_lapazMEX.precip.stream.txt 1 2004 2009 ',
                    'clm1PT.1x1pt_lapazMEX.other.stream.txt  1 2004 2009 ',
                    'presaero.stream.txt 1 2000 2000'
   vectors        = 'null','null','null','null'
   mapmask        = 'nomask','nomask','nomask','nomask'
   mapalgo        = 'nn','nn','nn','nn'
   tintalgo       = 'coszen','nearest','linear','linear'
   taxmode        = 'cycle','cycle','cycle','cycle'
  /
```

**Note:** The example above shows the resolved namelist and streams file after **cesm_setup** has been run.

We've outlined and given a few examples of using your own atmosphere forcing. In the next chapter we go into the details of using PTCLM1.110726.

# Chapter 6. Trouble Shooting Problems

In this chapter we give some guidance on what to do when you encounter some of the most common problems. We can't cover all the problems that a user could potentially have, but we will try to help you recognize some of the most common situations. And we'll give you some suggestions on how to approach the problem to come up with a solution.

In general you will run into one of three type of problems:

1. *setup-time*
2. *build-time*
3. *run-time*

You may also run into problems with **create_newcase** itself, or with the archiving scripts -- for those problems you should consult the CESM1.1.1 Scripts User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm).

## Trouble with Setup

The first type of problem happens when you invoke the **cesm_setup** command. This indicates there is something wrong with your input datasets, or the details of what you are trying to setup the model to do. There's also a trouble-shooting chapter in the CESM1.1.1 Scripts User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm). Many of the problems with configuration can be resolved with the guidelines given there. Here we will restrict ourselves to problems from the input files.

**Example 6-1. Example of cesm_setup problem with missing datasets**

```
> ./create_newcase -case ne60rcp6 -res ne60_g16 -compset IRCP60CN \
-mach yellowstone_intel
> ./cesm_setup
```

The following is what is displayed to the screen.

```
.
.
.
Running preview_namelist script
CLM configure done.
CLM adding use_case 1850-2100_rcp6_transient defaults for var clm_demand with val fpftdyn
CLM adding use_case 1850-2100_rcp6_transient defaults for var clm_start_type with val startup
CLM adding use_case 1850-2100_rcp6_transient defaults for var model_year_align_ndep with val 1850
CLM adding use_case 1850-2100_rcp6_transient defaults for var rcp with val 6
CLM adding use_case 1850-2100_rcp6_transient defaults for var sim_year with val 1850
CLM adding use_case 1850-2100_rcp6_transient defaults for var sim_year_range with val 1850-2100
CLM adding use_case 1850-2100_rcp6_transient defaults for var stream_year_first_ndep with val 1850
CLM adding use_case 1850-2100_rcp6_transient defaults for var stream_year_last_ndep with val 2100
CLM adding use_case 1850-2100_rcp6_transient defaults for var use_case_desc with val Simulate transient land-use, aerosol and Nitroge
with historical data from 1850 to 2005 and then with the RCP6 scenario from AIM

build-namelist - No default value found for fpftdyn.
            Are defaults provided for this resolution and land mask?
ERROR: clm.buildnml.csh failed
ERROR: /Users/erik/clm_cesm1_1_1_rel/scripts/ne60rcp6/preview_namelists failed: 25344
```

The important thing to note here is the line:

```
ERROR: clm.buildnml.csh failed
```

which tells us that the problem is in the land **clm.buildnml.csh**. It may also indicate problems in one of the other buildnml.csh files (atm, cesm, cpl, glc, ice, or ocn), in which case you should consult the appropriate model user's guide.

In the example, the error is that the CLM XML database does NOT have a `finidat` for the given resolution, rcp scenario and ocean mask. That means you will need to create the file and then supply the file into your case. See Chapter 2 for more information on creating files, and see Chapter 3 for more information on adding files to the XML database. Alternatively, you can provide the file to your case by creating a user namelist as shown in the Section called *User Namelist* in Chapter 1.

> **Note:** The two most common problems from your **clm.buildnml.csh** will be errors from the CLM **configure** or **build-namelist**. For more information on these scripts see: the Section called *More information on the CLM **configure** script* in Chapter 1 and the section on CLM_BLDNML_OPTS.

# Trouble with Building

Here's an example of running the build for a case and having it fail in the land model build. As you can see it lists which model component is being built and the build log for that component.

```
 CCSM BUILDEXE SCRIPT STARTING
 - Build Libraries: mct pio csm_share
Sat Jun 19 21:21:19 MDT 2010 /ptmp/erik/test_build/mct/mct.bldlog.100619-212107
Sat Jun 19 21:22:18 MDT 2010 /ptmp/erik/test_build/pio/pio.bldlog.100619-212107
Sat Jun 19 21:23:18 MDT 2010
/ptmp/erik/test_build/csm_share/csm_share.bldlog.100619-212107
Sat Jun 19 21:24:00 MDT 2010 /ptmp/erik/test_build/run/cpl.bldlog.100619-212107
Sat Jun 19 21:24:00 MDT 2010 /ptmp/erik/test_build/run/atm.bldlog.100619-212107
Sat Jun 19 21:24:06 MDT 2010 /ptmp/erik/test_build/run/lnd.bldlog.100619-212107
ERROR: clm.buildexe.csh failed, see /ptmp/erik/test_build/run/lnd.bldlog.100619-212107
ERROR: cat /ptmp/erik/test_build/run/lnd.bldlog.100619-212107
```

You can then examine the build log that failed and see what went wrong. Most compilers will give the full filepath and line number for the file that filed to compile.

# Trouble with Running

Tracking down problems while the model is running is much more difficult to do than setup or build problems. In this section we will give some suggestions on how to find run time problems. Below we show the log file results of a job that aborted while running.

```
 CCSM PRESTAGE SCRIPT HAS FINISHED SUCCESSFULLY
Sun Jun 20 18:24:06 MDT 2010 -- CSM EXECUTION BEGINS HERE
Sun Jun 20 18:24:35 MDT 2010 -- CSM EXECUTION HAS FINISHED
Model did not complete - see /ptmp/erik/test_run/run/cpl.log.100620-182358
```

In the next section we will talk about using the different log files to track down problems, and find out where the problem is coming from. In the section after that we give some general advice on debugging problems and some suggestions on ideas that may be helpful to track the problem down. Some of the examples below are from the `models/lnd/clm/doc/KnownBugs` (../KnownBugs) file.

## Tracking Problems by Querying Log Files

The first thing to do when tracking down problems is to query the different log files to see if you can discover where the problem occurs, and any error messages about it. It's important to figure out if the problem comes in at initialization or in the run phase of the model, and in which model component the problem happens. There are different log files for the different major components, and they all end with the date and time in YYMMDD-HHMMSS format (2-digit: year, month, day, hour minute and second). When the model runs to completion the log files will be copied to the `logs` directory in the script directory, but when the model fails they will remain in the run directory. Here's an example list of log files from an "I" case where the model dies in the land model initialization. For "I" cases the sea-ice and ocean components are just stubs and don't create log files (and unless running with the active land-ice model "glc" log files won't be created either).

```
atm.log.100620-182358
cesm.log.100620-182358
cpl.log.100620-182358
lnd.log.100620-182358
```

### The coupler log file

The first log file to check is the coupler log file so that you can see where the model dies and which model component it fails in. When the model dies at initialization the last model component listed is the component that failed.

Example of a case that fails in the CLM land model initialization.

```
(seq_timemgr_clockPrint)     Prev Time   = 00001201   00000
(seq_timemgr_clockPrint)     Next Time   = 99991201   00000
(seq_timemgr_clockPrint)     Intervl yms =     9999       0           0

(seq_mct_drv) : Initialize each component: atm, lnd, ocn, and ice
(seq_mct_drv) : Initialize atm component
(seq_mct_drv) : Initialize lnd component
```

### The cesm log file

The cesm log files are to some extent the "garbage collection" of log output. The CLM sends it's output from it's master processor, but sends other output and possibly errors to the cesm log file. Because, of this, often error messages are somewhere in the cesm log file. However, since there is so much other

output it may be difficult to find. For example, here is some output from an older version of CESM (CESM1.0.2) where the RTM river routing file (before it was converted to NetCDF) was not provided and the error on the open statement for the file was embedded near the end of the cesm log file.

```
NODE#  NAME
(    0)  be1105en.ucar.edu
"/gpfs/proj2/fis/cgd/home/erik/clm_trunk/models/lnd/clm/src/riverroute/RtmMod.F90", line
239: 1525-155 The file name provided in the OPEN statement for unit 1 has zero length or
contains all blanks.  The program will recover by ignoring the OPEN statement.
"/gpfs/proj2/fis/cgd/home/erik/clm_trunk/models/lnd/clm/src/riverroute/RtmMod.F90", line
241: 1525-001 The READ statement on the file fort.1 cannot be completed because the end
of the file was reached.  The program will stop.

Running: ./cesm.exe
Please wait...

Memory usage for   ./cesm.exe (task #   0) is:      51696 KB. Exit status: 1. Signal: 0
```

Although the example is from an earlier version of the model it still serves to illustrate finding problems from the cesm log file.

When working with the cesm log file, for a run-time problem, you will need to be able to separate it's output into three categories: pre-crash, crash, and post-crash. The pre-crash section is everything that is normal output for good operation of the model. The crash section is the section where the model dies and reports on the actual problem. the post-crash section is the cleanup and finalization after the model dies. The most important part of this of course is the crash section. The tricky part is distinguishing it from the other sections. Also because the cesm log file most likely has duplicated output from multiple processors it is even more difficult to distinguish the different sections and to some extent the sections may be intertwined, as different processors reach the different sections at different times. Because, of this reducing the number of processors for your simulation may help you sort out the output in the file (see the Section called *Run with a smaller set of processors*). Also much of the output from the cesm log file are system level information having to do with MPI multiprocessing. Usually you can ignore this information, but it makes it more difficult to trudge through.

Sometimes the cesm log file is the ONLY file available, because the model terminates early in initialization. In this case understanding the output in the cesm log file becomes even more important. This also indicates the model did NOT advance far enough to reach the initialization of the individual model components. This may mean that the initialization of the multiprocessing for MPI and/or OpenMP failed, or that the reading of the driver namelist file "drv_in" failed.

Here we show those three sections for a cesm log file where a two task job failed on reading the namelist file. For a typical job with many tasks similar sections of this will be repeated not just twice but for each task and hence make it harder to read.

*Pre-crash section of the cesm log file*

```
ATTENTION: 0031-386  MP_INSTANCES setting ignored when LoadLeveler is not being used.
ATTENTION: 0031-386  MP_INSTANCES setting ignored when LoadLeveler is not being used.
ATTENTION: 0031-378 MP_EUIDEVICE setting ignored when LoadLeveler is not being used.
ATTENTION: 0031-386  MP_INSTANCES setting ignored when LoadLeveler is not being used.
   0:INFO: 0031-724  Executing program: </usr/local/lsf/7.0/aix5-64/bin/lsnrt_run>
   1:INFO: 0031-724  Executing program: </usr/local/lsf/7.0/aix5-64/bin/lsnrt_run>
   0:/contrib/bin/cesm_launch: process 401894 bound to logical CPU 0 on host be0310en.ucar.edu ...
   1:/contrib/bin/cesm_launch: process 439264 bound to logical CPU 1 on host be0310en.ucar.edu ...
   0:INFO: 0031-619  64bit(us, Packet striping on)  ppe_rmas MPCI_MSG: MPI/MPCI library was compiled on   Wed Aug  5 13:36:06 2009
   0:
   1:LAPI version #14.26 2008/11/23 11:02:30 1.296 src/rsct/lapi/lapi.c, lapi, rsct_rpt53, rpt53s004a 09/04/29 64bit(us)  library comp
   1:.
   1:LAPI is using lightweight lock.
   0:LAPI version #14.26 2008/11/23 11:02:30 1.296 src/rsct/lapi/lapi.c, lapi, rsct_rpt53, rpt53s004a 09/04/29 64bit(us)  library comp
```

```
0:.
0:LAPI is using lightweight lock.
0:Use health ping for failover/recovery
1:Use health ping for failover/recovery
0:Initial communication over instance 2.
1:Initial communication over instance 0.
1:IB RDMA initialization completed successfully
1:The MPI shared memory protocol is used for the job
0:IB RDMA initialization completed successfully
0:LAPI job ID for this job is: 1684890719
0:The MPI shared memory protocol is used for the job
0:(seq_comm_setcomm)  initialize ID (  7 GLOBAL ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  2   ATM  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  1   LND  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  4   ICE  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  5   GLC  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  3   OCN  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_setcomm)  initialize ID (  6   CPL  ) pelist   =     0     1     1 ( npes =     2) ( nthreads =  1)
0:(seq_comm_joincomm) initialize ID (  8 CPLATM ) join IDs =     6     2       ( npes =     2) ( nthreads =  1)
0:(seq_comm_joincomm) initialize ID (  9 CPLLND ) join IDs =     6     1       ( npes =     2) ( nthreads =  1)
0:(seq_comm_joincomm) initialize ID ( 10 CPLICE ) join IDs =     6     4       ( npes =     2) ( nthreads =  1)
0:(seq_comm_joincomm) initialize ID ( 11 CPLOCN ) join IDs =     6     3       ( npes =     2) ( nthreads =  1)
0:(seq_comm_joincomm) initialize ID ( 12 CPLGLC ) join IDs =     6     5       ( npes =     2) ( nthreads =  1)
0:
0: (seq_comm_printcomms) ID layout : global pes vs local pe for each ID
0:     gpe         LND      ATM      OCN      ICE      GLC      CPL    GLOBAL   CPLATM   CPLLND   CPLICE   CPLOCN   CPLGLC    nthrds
0:     ---      ------   ------   ------   ------   ------   ------   ------   ------   ------   ------   ------   ------    ------
0:       0   :       0        0        0        0        0        0        0        0        0        0        0        0         1
1:       1   :       1        1        1        1        1        1        1        1        1        1        1        1         1
1:
0: (t_initf) Read in prof_inparm namelist from: drv_in
1: (seq_io_init) cpl_io_stride, iotasks or root out of bounds - resetting to defaults  4 0 1
0: piolib_mod.f90 1353 1 2 1 2
1: piolib_mod.f90 1353 1 2 1 2
0: pio_support::pio_die:: myrank= 0 : ERROR: piolib_mod.f90: 1354 : not enough procs for the stride
1: pio_support::pio_die:: myrank= 1 : ERROR: piolib_mod.f90: 1354 : not enough procs for the stride
```

### *Crash section of the cesm log file*

```
0:
0:  Traceback:
1:
1:  Traceback:
0:    Offset 0x00000c4c in procedure __pio_support_NMOD_piodie, near line 88 in file pio_support.F90.in
1:    Offset 0x00000c4c in procedure __pio_support_NMOD_piodie, near line 88 in file pio_support.F90.in
0:    Offset 0x00000fd0 in procedure __piolib_mod_NMOD_init, near line 1354 in file piolib_mod.F90
1:    Offset 0x00000fd0 in procedure __piolib_mod_NMOD_init, near line 1354 in file piolib_mod.F90
1:    Offset 0x00000398 in procedure __seq_io_mod_NMOD_seq_io_init, near line 247 in file /gpfs/proj2/fis/cgd/home/erik/clm_trunk/m
0:    Offset 0x00000398 in procedure __seq_io_mod_NMOD_seq_io_init, near line 247 in file /gpfs/proj2/fis/cgd/home/erik/clm_trunk/m
0:    Offset 0x0001aa88 in procedure ccsm_driver, near line 465 in file /gpfs/proj2/fis/cgd/home/erik/clm_trunk/models/drv/driver/c
0:    --- End of call chain ---
1:    Offset 0x0001aa88 in procedure ccsm_driver, near line 465 in file /gpfs/proj2/fis/cgd/home/erik/clm_trunk/models/drv/driver/c
1:    --- End of call chain ---
```

### *Post-crash section of the cesm log file*

```
1:Communication statistics of task 1 is associated with task key: 1684890719_1
0:Communication statistics of task 0 is associated with task key: 1684890719_0
0:
0:Running: ./cesm.exe
0:Please wait...
0:
0:Memory usage for   ./cesm.exe (task #   0) is:     198892 KB. Exit status: 134. Signal: 0
1:
1:Running: ./cesm.exe
1:Please wait...
1:
```

```
   1:Memory usage for   ./cesm.exe (task #   0) is:     198572 KB. Exit status: 134. Signal: 0
INFO: 0031-656  I/O file STDOUT closed by task 0
INFO: 0031-656  I/O file STDERR closed by task 0
ERROR: 0031-250  task 0: IOT/Abort trap
INFO: 0031-656  I/O file STDOUT closed by task 1
INFO: 0031-656  I/O file STDERR closed by task 1
ERROR: 0031-250  task 1: IOT/Abort trap
INFO: 0031-639  Exit status from pm_respond = 0
ATTENTION: 0031-386  MP_INSTANCES setting ignored when LoadLeveler is not being used.
Job  /usr/local/lsf/7.0/aix5-64/bin/poejob /contrib/bin/ccsm_launch /contrib/bin/job_memusage.exe ./cesm.exe

TID   HOST_NAME   COMMAND_LINE            STATUS            TERMINATION_TIME
===== ========== ================  ======================  ===================
00000 be0310en   /contrib/bin/ccs  Exit (134)              08/31/2010 12:32:57
00001 be0310en   /contrib/bin/ccs  Exit (134)              08/31/2010 12:32:57
```

## The CLM log file

Of course when you are working with and making changes to CLM, most of your focus will be on the CLM log file and the errors it shows. As already pointed out if you don't see errors in the `lnd.log.*` file you should look in the `cesm.log.*` to see if any errors showed up there.

Here's an example of the `lnd.log.*` file when running PTS_MODE with initial conditions (this is bug 1025 in the `models/lnd/clm/doc/KnownLimitationss` (../KnownLimitations) file).

```
 Successfully initialized variables for accumulation

 reading restart file I2000CN_f09_g16_c100503.clm2.r.0001-01-01-00000.nc
 Reading restart dataset
 ERROR - setlatlon.F:Cant get variable dim for lat or lsmlat
 ENDRUN: called without a message string
```

## The DATM log file

When working with "I cases" the second most common problems after CLM problems are problems with the data atmosphere model. So examining the `atm.log.*` is important.

Here's an example of a problem that occurs when the wrong prescribed aerosol file is given to a `pt1_pt1` simulation.

```
(datm_comp_init)  atm mode = CLMNCEP
(shr_strdata_init)  calling shr_dmodel_mapSet for fill
(shr_strdata_init)  calling shr_dmodel_mapSet for remap
 ('shr_map_getWts') ERROR: yd outside bounds  19.5000000000000000
(shr_sys_abort) ERROR: ('shr_map_getWts')  ERROR yd outside 90 degree bounds
(shr_sys_abort) WARNING: calling shr_mpi_abort() and stopping
```

### The batch log files

The names of the batch log files will depend on the batch system of the machine that is being used. They will normally be in the script directory. Usually, they don't contain important information, but they are a last resort place to look for error messages. On the NCAR system "yellowstone" the batch files are called with names that start with the batch submission script and then either "stderr.o" or "stdout.o", with the job number at the end.

## General Advice on Debugging Run time Problems

Here are some suggestions on how to track down a problem while running. In general if the problem still occurs for a simpler case, it will be easier to track down.

1. *Run in DEBUG mode*
2. *Run with a smaller set of processors*
3. *Run in serial mode with a single processor*
4. *Run at a lower resolution*
5. *Run a simpler case*
6. *Run with a debugger*

### Run in DEBUG mode

The first thing to try is to run in DEBUG mode so that float point trapping will be triggered as well as array bounds checking and other things the compiler can turn on to help you find problems. To do this edit the `env_build.xml` file and set DEBUG to `TRUE` as follows:

```
> ./xmlchange DEBUG=TRUE
```

### Run with a smaller set of processors

Another way to simplify the system is to run with a smaller set of processors. You will need to clean the setup and edit the `env_mach_pes.xml`. For example, to run with four processors:

```
> ./cesm_setup -clean
> ./xmlchange NTASKS_ATM=4,NTASKS_LND=4,NTASKS_ICE=4,NTASKS_OCN=4,NTASKS_CPL=4,NTASKS_GLC=4
> ./cesm_setup
```

Another recommended simplification is to run without threading, so set the NTHRDS for each component to "1" if it isn't already. Sometimes, multiprocessing problems require a certain number of processors before they occur so you may not be able to debug the problem without enough processors.

But, it's always good to reduce it to as low a number as possible to make it simpler. For threading problems you may have to have threading enabled to find the problem, but you can run with 1, 2, or 3 threads to see what happens.

## Run in serial mode with a single processor

Simplifying to one processor removes all multi-processing problems and makes the case as simple as possible. If you can enable MPILIB=mpi-serial you will also be able to run interactively rather than having to submit to a job queue, which sometimes makes it easier to run and debug. If you can use MPILIB=mpi-serial you can also use threading, but still run interactively in order to use more processors to make it faster if needed.

```
> ./cesm_setup -clean
# Set tasks and threads for each component to 1
# You could also set threads to something > 1 for speed, but still
# run interactively if threading isn't an issue.
> ./xmlchange NTASKS_ATM=1,NTHRDS_ATM=1,NTASKS_LND=1,NTHRDS_LND=1,NTASKS_ICE=1,NTHRDS_ICE=1
> ./xmlchange NTASKS_OCN=1,NTHRDS_OCN=1,NTASKS_CPL=1,NTHRDS_CPL=1,NTASKS_GLC=1,NTHRDS_GLC=1
# set MPILIB to mpi-serial so that you can run interactively
> ./xmlchange MPILIB=mpi-serial
> ./cesm_setup
# Then build your case
# And finally run, by running the *.run script interactively
```

## Run at a lower resolution

If you can create a new case running at a lower resolution and replicate the problem it may be easier to solve. This of course requires creating a whole new case, and trying out different lower resolutions.

## Run a simpler case

Along the same lines, you might try running a simpler case, trying another compset with a simpler setup and see if you can replicate the problem and then debug from that simpler case. Again, of course you will need to create new cases to do this.

## Run with a debugger

Another suggestion is to run the model with a debugger such as: **dbx**, **gdb**, or **totalview**. Often to run with a debugger you will need to reduce the number of processors as outlined above. Some debuggers such as **dbx** will only work with one processor, while more advanced debuggers such as **totalview** can work with both MPI tasks and OMP threads. Even simple debuggers though can be used to query core files, to see where the code was at when it died (for example using the **where** in **dbx** for a core file can be very helpful. For help in running with a debugger you will need to contact your system administrators for the machine you are running on.

# Chapter 7. Scripts for testing CLM

Technically, you could use the customization we gave in Chapter 1 to test various configuration and namelist options for CLM. Sometimes, it's also useful to have automated tests though to test that restarts give exactly the same results as without a restart. It's also useful to have automated tests to run over a wide variety of configurations, resolutions, and namelist options. To do that we have several different types of scripts set up to make running comprehensive testing of CLM easy. There are two types of testing scripts for CLM. The first are the CESM test scripts, which utilize the **create_newcase** scripts that we shown how to use in this User's Guide. The second are a set of stand-alone scripts that use the CLM **configure** and **build-namelist** scripts to build and test the model as well as testing the CLM tools as well. Below we will go into further details of how to use both methods.

## Testing CLM Using the CESM Test Scripts

We first introduce the test scripts that work for all CESM components. We will use the **create_test** and then the **create_test_suite** scripts. The **create_test** runs a specific type of test, at a given resolution, for a given compset using a given machine. There is a list of different tests, but the "ERI" tests do several things at once, running from startup, as well as doing exact branch and restart tests. So to run "ERI" testing at 2-degree with the I1850CN compset on yellowstone_intel you do the following.

```
> cd scripts
> ./create_test -testname ERI.f19_g16.I1850CN.yellowstone_intel
> cd ERI.f19_g16.I1850CN.yellowstone_intel.$id
> ./ERI.f19_g16.I1850CN.yellowstone_intel.$id.build
> ERI.f19_g16.I1850CN.yellowstone_intel.$id.submit
```

When the test is done it will update the file `TestStatus` with either a PASS or FAIL message.

To run a suite of tests from a list of tests with syntax similar to above you use **create_test_suite** as follows passing it a ASCII list of tests. There are already some test lists in the `scripts/ccsm_utils/Testlists` directory a few of which are specific to CLM. To run for the CLM yellowstone test list, on yellowstone, you would do the following:

```
> cd scripts
> ./create_test_suite -input_list ccsm_utils/Testlists/yellowstone.clm.auxtest
# Submit the suite of tests (note $id refers to the integer job number for this job)
> ./cs.submit.$id.yellowstone
# Later check the tests with...
> ./cs.status.$id
# The above will give a PASS or FAIL message for each test.
```

For more information on doing testing with the CESM scripts see the CESM1.1.1 User's Guide (http://www.cesm.ucar.edu/models/cesm1.1/cesm) on testing.

## Testing CLM Using the CLM Specific Stand-Alone Tools Testing Scripts

# Testing CLM tools using the CLM Stand-Alone Tools Testing Scripts

In the `models/lnd/clm/test/system` directory there is a set of test scripts that is specific to stand-alone CLM tools. It allows you to test the CLM tools such as **mkgriddata** and **mksurfdata_map**. The main driver script is called `test_driver.sh` and it is normally run interactively. Like other scripts you can get help on it by running the "-help" option as: **test_driver.sh -help**. There is also a `README` file that gives details about environment variables that can be given to **test_driver.sh** to change it's operation.

To run tests interactively:

```
> cd models/lnd/clm/test/system
> ./test_driver.sh -i
```

The output of the help option is as follows:

```
Only setup to work on: yellowstone, bluefire, mirage, lynx, jaguarpf, edinburgh, and yong
```

A table of the list of tests and the machines they are run on is available from: test_table.html (../../test/system/test_table.html)

# Appendix A. Building the Users-Guide Documentation for CLM

All of the documentation for CLM can be built using GNU Makefiles that are available in the appropriate directories. The Makefiles require the following utilities: **docbook2html**, **docbook2pdf**, **protex**, and **latex2html**.

To build the Users Guide for CLM (requires docbook).

```
> cd models/lnd/clm/doc/UsersGuide
> gmake
```

Note, that when the Users-Guide is built it will get output from other CLM utilities that by nature abort, and hence stop the make from continuing. However, this is expected so you should simply run **gmake** again until it either completes or comes upon a legitimate issue. Here is what a sample warning looks like when **gmake** is run.

```
The following line will fail in the make as it calls die -- but that is expected
Check that the output config_help.tlog is good and redo your make
../../bld/configure -help >&` config_help.tlog
make: *** [config_help.tlog] Error 255
```

To build the Code Reference Guide for CLM (requires **protex** and **latex2html**). The make here uses a `Filepath` file that points to the list of directories that you want **protex** to run over. You should examine this file and make sure it is appropriate for what you need to do, before running the make.

```
> cd models/lnd/clm/doc/CodeReference
> gmake
```

To build the table of tests for the CLM test suite. The make here runs a UNIX shell script to create a html table of the list of tests run on the different machines from the CLM test suite.

```
> cd models/lnd/clm/test/system
> gmake
```