Odycloud Numerical Weather Prediction & Air Quality Modeling on AWS

'Odycloud Numerical Weather Prediction & Air Quality Modeling on AWS' is an integrated AMI that includes up-to-date executables for WRF and CMAQ along with several tools able to download files of interest (e.g. GFS data) and postprocess the results. Version 5.1.0 includes precompiled variations of WRFv4.6 (ARW, WRF-Chem, WRFDA, and WRFPLUS), and CMAQ v5.5 to perform air quality analysis complementing weather forecasts. This User's Guide provides instructions on how to launch EC2 instances and clusters running WRF and CMAQ. The AMI works either with single instances or in conjunction with AWS ParallelCluster for cluster launch. For users interested in working with a Linux desktop environment, the AMIs already include the extra elements and open the necessary ports; section 6 provides further details on how to create graphical sessions. Modelers interested in a full graphical experience able to control all aspects of modeling via a GUI interface might consider using 'GUI for Numerical Predictions in the public cloud (Graviton3 & Graviton4).'

Odycloud support

Subscription to the AMI includes support from Odycloud on how to use the AMI with AWS infrastructure and how to set up HPC clusters. We also help customers to get started with some specific simulations if needed. AWS offers a very large variety of services, which grow almost on a weekly basis. Many of our customers who are new to cloud environments and AWS use videoconferencing to get extra support. If either you are getting familiar with the AWS environment or wish to learn how to use the AMIs, feel free to contact us (support@odyhpc.com) to schedule any such session or for any other questions.

March -2025

TABLE OF CONTENTS

Instances

1. Launching EC2 instances

The following instructions discuss how to launch EC2 instances running WRF and CMAQ.

a. **Account creation** - 'Odycloud Numerical Weather Prediction & Air Quality Modeling on AWS' can be used by any user with an active AWS account. If you are new to AWS, follow the instructions at https://aws.amazon.com/premiumsupport/knowledge-center/create-and-activate-aws-account/ to create a new account. You will also need to create an IAM user (https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_create.html) and assign the required permissions to create keypairs and launch instances (https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_change-permissions.html). You do not need the access & secret keys for running WRF on a single instance, but you will need them for more complex actions such as cluster launch or downloading data from S3 buckets. Either way, keep these keys safely as required by AWS.

b. **Subscribing to the AMI** - The AMI with the preinstalled software can be downloaded from the AWS Marketplace (https://aws.amazon.com/marketplace/pp/prodview-4axaf4xxxa6te). Make sure to understand the charges for AWS infrastructure and for the AMI.

c. **Launching instances** - Once your subscription is active, you can launch available instances based on your choices of architecture and configuration. More information about launching EC2 instances is available at https://docs.aws.amazon.com/quickstarts/latest/vmlaunch/step-1-launch-instance.html. The AMI is available in most regions and AZs. A few tips for launching instances follow:

   i. Spot instances are available per the usual conditions.

   ii. The AMI weighs 125 GiB but has little free space. Most production runs will require far more free space; we and some of our customers have used disk space exceeding 5 TB. AWS makes it easy to handle large storage needs and users have several options. Two easy options are to increase disk storage at the time of launching instances or to use EBS volumes. The latter can be particularly useful for large datasets to be recycled with different instances and clusters. Section 2 describes these options in greater detail.

   iii. Choose the SG (Security Group) according to your own needs. As a minimum, it should have port 22 open. This is the default configuration.

iv. Once you have selected the region, you will need to use your own existing keypair or create a new one for launching WRF.

d. **Connection to the instance** – In order to connect to your instance, you must use a SSH client such as PuTTY, MobaXterm or a similar app (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html). Your username is 'ubuntu' and you enjoy superuser privileges.

e. **Quick guide to running WRF** – The WPS and WRF apps are installed at the /home/ubuntu/PREPRO/WPS and /home/ubuntu/WRF-4.6 subdirectories, respectively. Preparing the case files can be completed either with the pre-installed WPS or by directly transferring the input files. After the files are ready, WRF is run with $ mpirun -np N wrf.exe where N is the number of MPI ranks. If the case files and the executable are not in the same directory, you will need to either create a symlink with the file or include the subdirectory when invoking WRF (/home/ubuntu/WRF-4.6/main/wrf.exe). We recommend running one MPI rank per core (e.g. it would be mpirun -np 64 ./wrf.exe for a c7i.32xlarge instance) but running in hyperthreaded mode requires adding the '–oversubscribe' flag' (e.g. mpirun –oversubscribe -np 128 ./wrf.exe).

f. **Other precompiled apps** - In addition to the latest WRF (ARW) release, the AMI incorporates precompiled executables for several WRF related apps:

i. WRF-Chem with kinetic pre-processor at /home/ubuntu/WRF-Chem.

ii. WRF Data Assimilation (WRFDA) at /home/ubuntu/DA/WRFDA.

iii. WRFPLUS at /home/ubuntu/DA/WRFPLUS.

iv. WRF-v3 is available per request (support@odyhpc.com).

2. Storage options

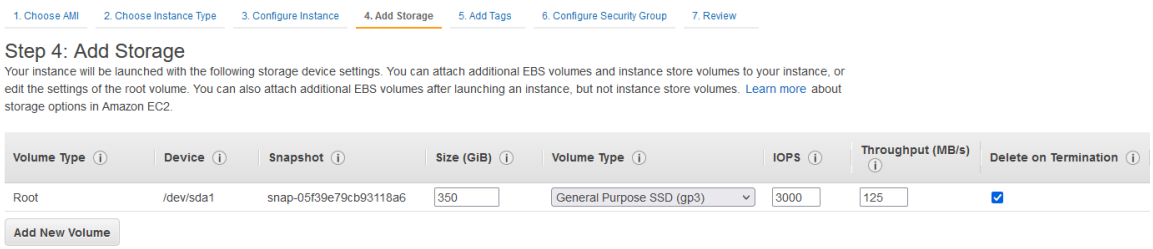
One of the advantages of using AWS infrastructure is the availability of many storage solutions meeting different needs and criteria. It is not the purpose of this section to cover every available option, which has its own voluminous documentation, but to describe a few options of interest to model users. The two most common options are EBS (Elastic Block Store) and S3 (Simple Storage Service). This section covers how to use the AMI with these two types of storage options, which should meet most of the AMI users' demands. Two more advanced options include the use of Lustre filesystems for high performance I/O attached to clusters, and Amazon EFS (Elastic File System), which can provide a general filesystem attached to several instances or clusters. We strongly advise to read the specialized literature if you are planning to use either Lustre or EFS. You are also welcomed to contact us if you have any question about how to proceed with these filesystem types.

a. **Storage attached to EC2 instances** - Launching an EC2 instance requires attaching some disk space, which AWS calls EBS. Any OS is automatically imprinted in this format at the time of instance creation, and so do the apps and dependencies contained in the AMI. This disk space constitutes the root volume associated to the instance. Any OS or AMI from the Marketplace has a predetermined volume that constitutes the minimum size of the root volume. However, our AMIs are usually listed with minimal free space and one of the first tasks is usually to increase disk space unless running some of the prepackaged examples in the AMI. When launching an instance, increasing the available disk space can be accomplished either through increasing the root volume size or by attaching another EBS volume to the instance. We discuss the former first before turning to the second.

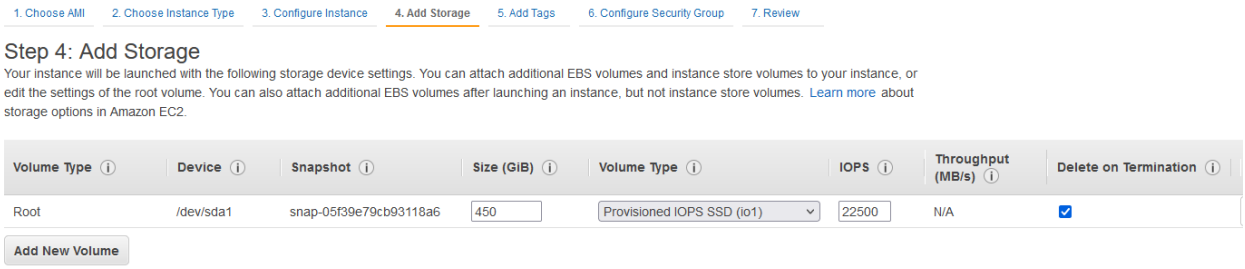
    i. Increasing the root volume attached to the EC2 instance

    Launching an EC2 instance from the console is a 7-step process although steps 3-6 can be skipped for a quick launch. After selecting the AMI, instance type and some instance details, step 4 offers the option to customize the root storage attached to the instance. As previously mentioned, it is a good idea to select a different type of EBS type than the general purpose (gp2) chosen by default. A discussion on the technical specifications of different EBS types is available, for example, at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html. Furthermore, this is a convenient time to increase the root

volume combining both steps. The first example seeks to run WRF using a 350 GiB disk on a gp3 SSD.

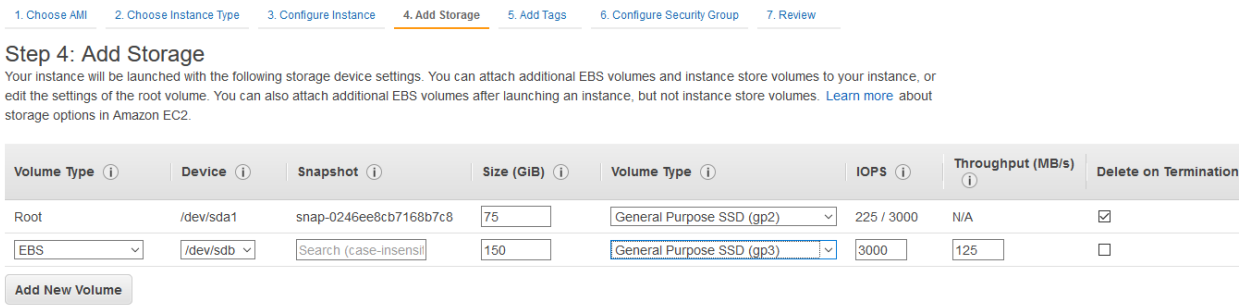


The next example is similar but uses a 450GiB io1 SSD for running CMAQ



ii. Attaching a second volume to the EC2 instance

In the next example, we are adding 150 GB of disk space with the caveat that our choice is of the type gp3. The advantage of using this extra EBS volume (instead of simply increasing the root volume) is that the data in this filesystem can be used separately from the instance including its attachment to other instances or clusters.

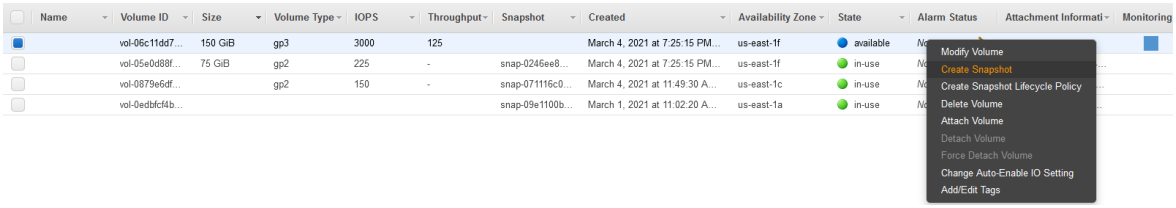


The extra space is not immediately available, but it needs to be mounted. An example of how to mount this EBS volume on the data subdirectory follows:

```
$ lsblk                          ! To check the name of the volume
$ sudo file -s /dev/nvme1n1
$ sudo mkfs -t xfs /dev/nvme1n1
```

```
$ sudo mkdir data
$ sudo mount /dev/nvme1n1 data
$ sudo chown ubuntu: ubuntu data
```

After creating or running any case, and to reuse or store this block, we can simply unmount it `$ sudo umount -d /dev/nvme1n1`, detach the volume and create a snapshot from the console volume menu.



Also, it is important to notice that we did not select 'Delete on Termination' previously so terminating the instance will not delete the volume and the meter will keep running until its deletion from the console or alternatively the CLI.

b. **Long-term storage** - S3 is more adequate for long-term storage, backup, and access to large databases. S3 organizes all the data into S3 buckets (they can also be seen as folders, but AWS always uses the word 'bucket'), which can be kept private or made public. There is plenty of online discussions about AWS S3, plus the official documentation (https://docs.aws.amazon.com/s3/index.html) can serve as a starting point. As usual, AWS documentation is voluminous and many model users might prefer some quick start guide.
As the apps included in the AMI generate huge amounts of data, S3 is an excellent tool for storing files, especially for long periods of time. Keep in mind that transferring data from an instance to a bucket is most of the time free (check current AWS documentation for up-to-date fares) but storing data or transferring files to on-premises machines will incur charges. Another fact to keep in mind is that it is not possible to store folders in S3 buckets (only files are accepted) so you might need to tar and compress subdirectories if the goal is to store them. To access non-public buckets belonging to your organization from an EC2 instance, it is necessary to configure your credentials with `$ aws configure` and input the IAM credentials (aws_access_key_id and aws_secret_access_key).

3. Running WRF simulations

Running a WRF simulation with the AMI works very similarly to any other Linux system. Once the instance has finished spinning up, you can download the meteorological data (see section 4 for more information) and use the WPS preprocessing apps (geogrid, ungrib, and metgrid) before performing any WRF simulations.

a. **WPS and ARW** – The WPS tools and ARW solver are installed at the `/home/ubuntu/PREPRO/WPS` and `/home/ubuntu/WRF-4.6/test/em_real` subdirectories, respectively. Preparing the case files can be completed either with the pre-installed WPS or by directly transferring the input files. After the files are ready, WRF is run with $ mpirun -np N wrf.exe where N is the number of MPI ranks. If the case files and the executable are not in the same directory, you will need to either create a symlink with the file or include the subdirectory when invoking WRF (`/home/ubuntu/WRF-4.6/test/em_real/wrf.exe`). Something to keep in mind when running HPC simulations is memory requirements. WRF is not particularly memory hungry compared to other HPC apps. The memory requirements can fluctuate slightly depending on the number of MPI ranks; for the test case, it requires a minimum of 10 GiB, which makes it suitable to run on single instances with 16 GiB of memory or higher. If memory is insufficient, the simulation will stop with an error message like:

```
--------------------------------------------------------------------------
Primary job  terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
--------------------------------------------------------------------------
--------------------------------------------------------------------------
mpirun noticed that process rank 2 with PID 0 on node ip-172-31-3-9 exited on signal 9 (Killed).
--------------------------------------------------------------------------
```

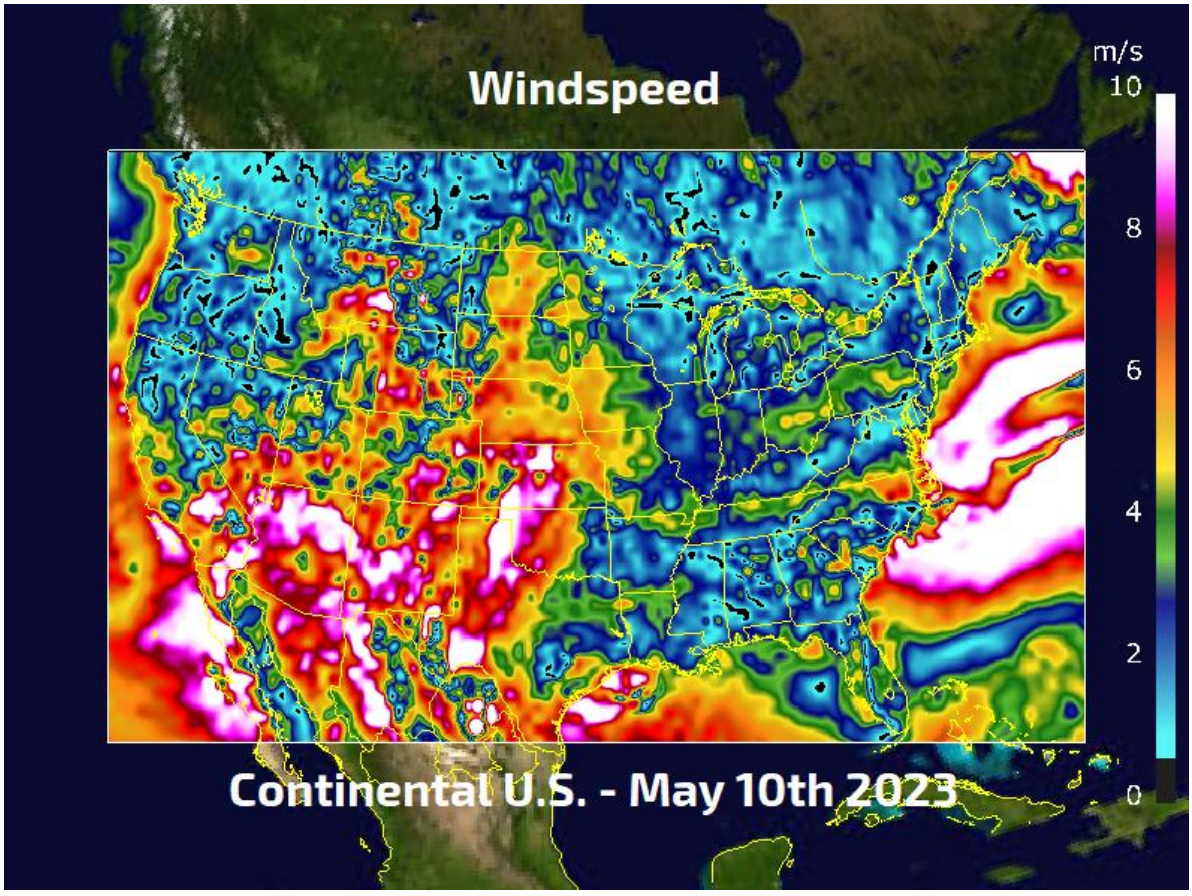
If memory is sufficient and everything else goes without a problem, the app will complete the simulation and the output files will be in the `/home/ubuntu/WRF-4.6/test/em_real` subdirectory unless the user specifically modifies *namelist.input* to reroute them to a different subdirectory.

b. **Running the test case** – The AMI comes with a test case consisting of a parent domain covering the continental U.S. and a nested domain covering the Washington D.C. area. This case has been previously preprocessed and the initial and lateral boundary condition files (wrfinput_d01, wrfinput_d02 and

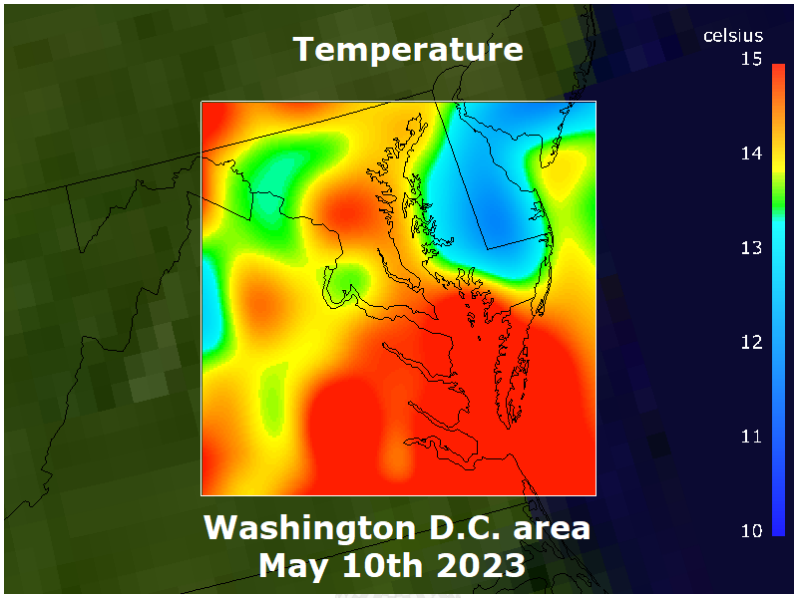
wrfbdy_d01) are also located at the /home/ubuntu/WRF-4.6/test/em_real subdirectory. The test-case, which can be used either as it-is or slightly modified, can help users to either get acclimated to the AWS environment and the AMI itself. Running the test case as-it-is simply requires moving into the subdirectory where the initial and boundary conditions are and invoking the app:

```
$ cd /home/ubuntu/WRF-4.6/test/em_real
$ mpirun -np N ./wrf.exe  (where N represents the number of MPI ranks)
```

The test case has a major domain covering the continental U.S. with a grid of 611 by 371 points and a resolution of 8 x 8 km^2. The vertical resolution is 33 points and the time resolution for this major domain is 45 s. The data are generated for a period of 48 hours starting on May 10, 2023.



The nested domain of the test case centers around Washington D.C. and covers most of states of Maryland, Virginia, and Delaware and a small of portion of New Jersey, Pennsylvania, and West Virginia. The grid is 200 by 200 points with a resolution of 1,600 x 1,600 m^2 (approximately one square mile). The time-step for this nested domain is 9 s.



c. **Other precompiled apps** - In addition to the latest WRF (ARW) release, the AMI incorporates precompiled executables for several WRF related apps:

   i. WRF-Chem with kinetic pre-processor at /home/ubuntu/WRF-Chem

   ii. WRF Data Assimilation (WRFDA) at /home/ubuntu/DA/WRFDA

   iii. WRFPLUS at /home/ubuntu/DA/WRFPLUS

   Performing simulations with these variations involves tailoring the *namelist* files according to the type of simulation to be performed. Specific information about how to perform simulations with WRF v3 or NMM is described in section 8.

4. Data download, preprocessing, and postprocessing tools

In addition to the precompiled WRF executables, the AMI includes several pre and postprocessing tools. Because there is a constant flow of new apps and versions, feel free to contact us if you are interested in a particular one even if it is not discussed in the present document.

a. **Data download** -There are many different sources of meteorological data as discussed in the WRF documentation. The most common meteorological data are GFS and NAM, with the latter being mostly used by U.S. organizations. As downloading GFS data, particularly from the NCEI dataset, is very time consuming, the AMI includes several scripts at `/home/ubuntu/DATA` facilitating the download of GFS and NAM files in an accelerated fashion.
   1 – GFS_download.exe will sequentially ask you for the GFS resolution, start date, number of days and if you want to download the first dataset for the following day before starting the download. This script will download the 4 datasets (00, 06, 12, 18) for each selected day plus the first (00) for the following day if requested.
   2 – download_NAM.exe works similarly to GFS_download but it downloads NAM files instead of GFS ones.
   3 – GFS_batch.exe downloads the GFS data based on the arguments to the batch file without having to enter any data interactively. The call to the batch file must follow the following syntax:
   `$ /home/ubuntu/DATA/GFS_batch.exe -r XXX -s YYYYMMDDHH -e YYYYMMDDHH`
   Here, 'r' is the GFS data resolution (0.25/0.5) where valid entries are 025, 05 and 050, 's' is the starting date that must follow the format (YYYYMMDDHH) and 'e' is the end date. Hours (HH) must be 00, 06, 12 or 18. The GFS files using either script will be stored at `/home/ubuntu/DATA/GFS` based on dates. The download of NAM or other meteorological data must follow standard procedures; contact us if you have any questions about it.
b. **Preprocessing tools** - The subdirectory `/home/ubuntu/PREPRO/WPS` is set up to perform the usual preprocessing steps before running WRF. For example, and once the data is loaded in the appropriate directory and the namelist.wps file is modified to reflect the target parameters for the simulation, the sequence of preprocessing commands would read:

```
$ cd /home/ubuntu/PREPRO/WPS
$ ./geogrid.exe
$ ln -sf /home/ubuntu/PREPRO/WPS/ungrib/Variable_Tables/Vtable.GFS
```

```
/home/ubuntu/PREPRO/WPS/Vtable
[$ ln -sf /home/ubuntu/PREPRO/WPS/ungrib/Variable_Tables/Vtable.NAM or alternative
/home/ubuntu/PREPRO/WPS/Vtable] for NAM or alternative data
$ ./link_grib.csh /home/ubuntu/DATA/GFS/gfs_3_2023
[$ ./link_grib.csh /home/ubuntu/DATA/NAM/nam_218_2023]
$ ./ungrib.exe
$ ./metgrid.exe
```

Once the met_em… files are generated, they can be either directly copied to the WRF subdirectory (e.g. `/home/ubuntu/WRF-4.6/test/em_real`) or symlinks can be used.

c. Postprocessing tools - The AMI also incorporates several postprocessing apps. This subsection outlines the available apps and any significant detail in their integration on the AMI. However, it does not describe them, and the modeler is referred to the app documentation.

i. **WRF-python** can be used as in any other Linux system.
ii. **ARWpost** is available at `/home/ubuntu/POSTPRO`. Once the WRF simulation is complete, ARWpost can be used to convert the output files to a different file format.
iii. **GrADS** is also available at `/home/ubuntu/POSTPRO`. Starting to process some data simply requires invoking the app from a graphical environment:

```
$ cd /home/ubuntu/POSTPRO/grads-2.2.2
$ ./grads
```

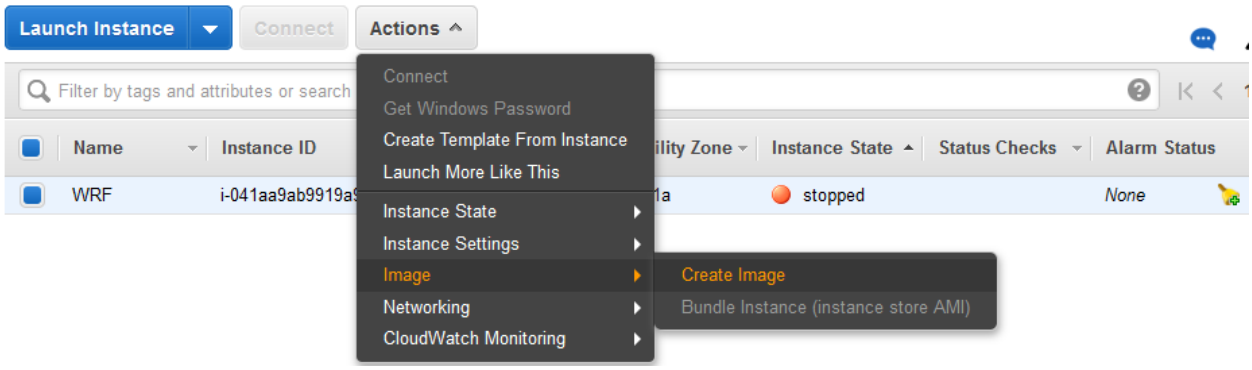
The NCAR Team strongly recommends WRF-python for postprocessing tasks, and we second this approach as a modern one, while providing greater flexibility than older apps.

5. Tailoring your AMI

The AMI is built with the latest stable version of the app and with the case discussed in section 3. However, this is a relatively simple case, and most users will want to compute more complex cases. It is possible to use the provided WPS to build the files necessary for a new case or to simply transfer them. If this new case is going to be reused several times or you wish to store it long-term, we strongly recommend using AWS EBS (Elastic Block Storage) and AWS S3 (Simple Storage Service), respectively.

Before launching the instance and building the new case, it is important to assess how much space will be needed. Most of our custom AMIs have little free space but, at the time of launching an instance, it is feasible to increase the available space from the 'Launch an instance' menu of the AWS console. You can select between increasing the size of the root volume or adding a new volume. The former is easy to select, and no further step is needed, whereas the latter requires mounting a new volume but has the advantage of dedicated EBS to that case (see the storage options section for more details).

Once the new case is available and any other tailoring is complete, you can build your own AMI by clicking on 'Creating Image' as shown in the screenshot or using the CLI.



The new AMI is available only to IAMs in your account. It has the same features as the original AMI including its availability for launching clusters. However, and unlike the original AMI, it is available only in the region where created.

6. Using a Linux desktop environment

WRF and CMAQ users exploring the option of migrating to the public cloud cite the lack of a graphical environment as one of the major drawbacks associated with this option. To overcome this challenge with a simple procedure, the AMIs are fully ready to start a Linux desktop environment session. Launching the AMI will also open the necessary ports so that users do not need to modify SGs or worry about other networking setups.

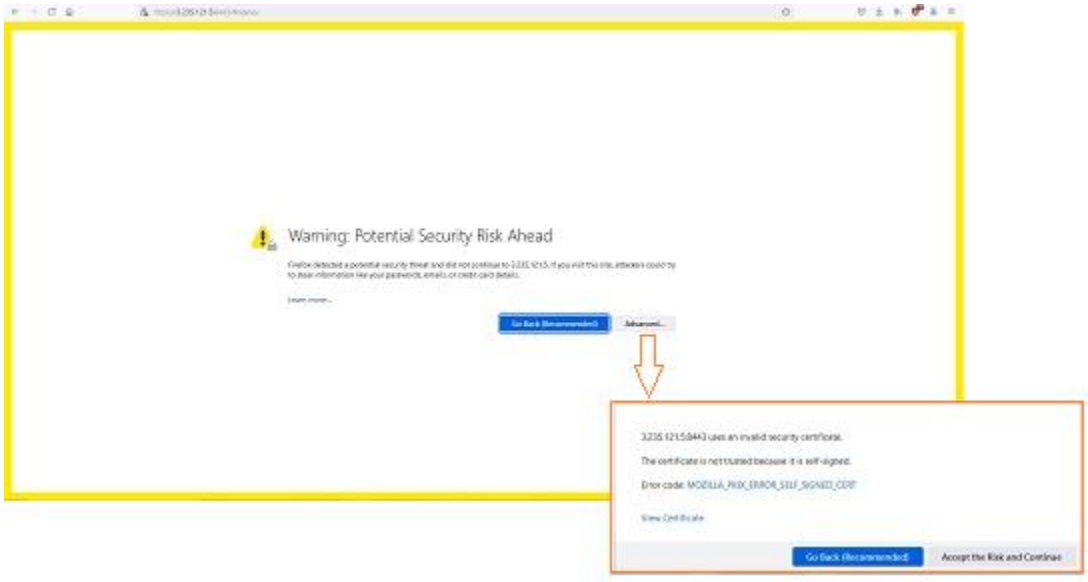
The AMIs utilize a high-performance display app named NICE-DCV (https://docs.aws.amazon.com/dcv/index.html) and currently owned by AWS. In the case of single instances, the only steps to initialize the desktop environment are to start the server and initialize the session[1] from a terminal:

```
$ sudo systemctl start dcvserver
$ dcv create-session custom_name_for_your_session
```

To check if the session is running and its characteristics, you can simply type:

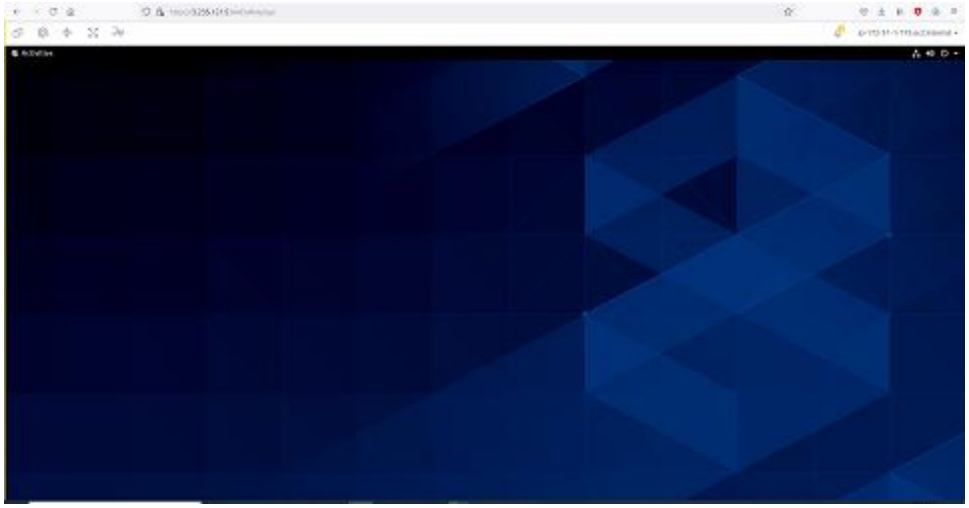
```
$ dcv list-sessions
$ dcv create-session custom_name_for_your_session
```

The session will be available in most web browsers at https://External_IP_for_your_instance:8443/#custom_name_for_your_session.
The first 2 screens will inform you of a potential certificate issue; this is normal as the app uses a self-certificate with NGINX.

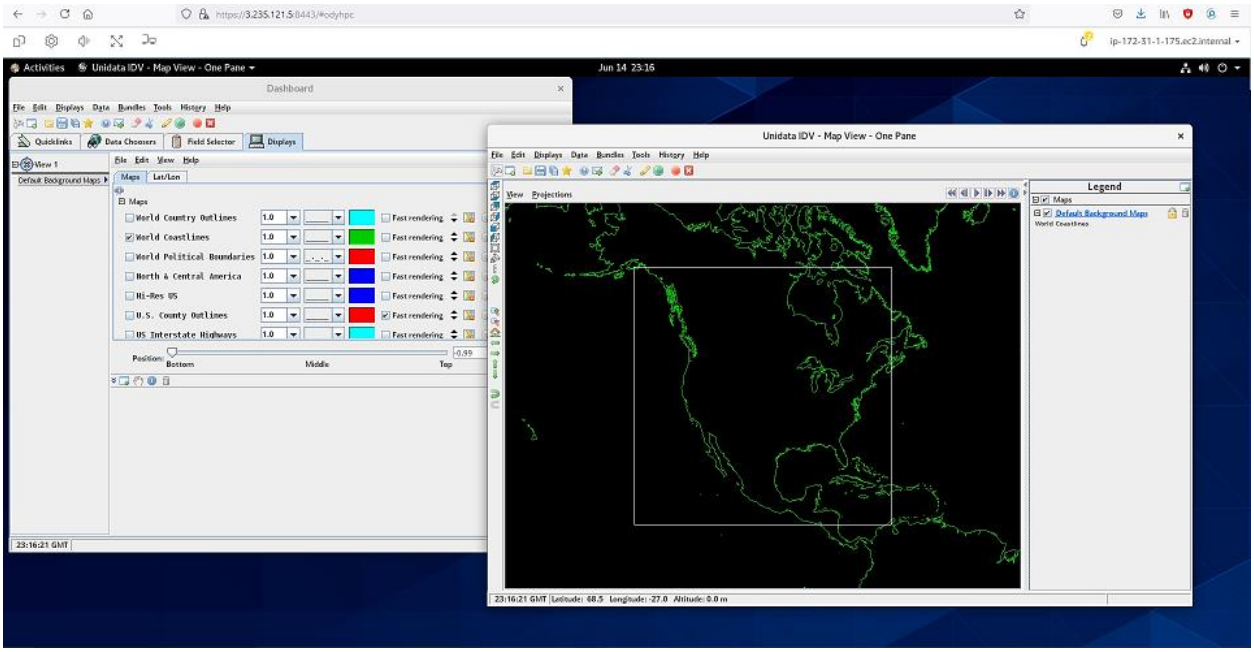


[1] Graphical sessions consume resources quickly, particularly memory. Although it is possible to start a session with 4 GiB of memory, we recommend using an instance with a minimum memory of 8GiB or even more if any visualization app is going to be used.

After clicking on 'Accept Risk and Continue,' the GNOME welcome screen will appear:



If, for example, IDV is installed and opened, the app can be used as with any other graphical desktop:



Modelers interested in a full graphical experience able to control all aspects of modeling from domain creation to postprocessing tools might consider using 'GUI for Numerical Predictions in the public cloud (Graviton3 & 3e)' that includes both WRF and CMAQ.

7. Running CMAQ

The AMI comes with CMAQ v5.5 precompiled at the subdirectory:

- /home/ubuntu/CMAQ/CCTM/scripts/BLD_CCTM_v55_gcc

Unlike WRF, the preferred method to launch CMAQ jobs is with the aid of scripts, which describe the computational parameters as well as the physical details of the computations to be performed. CMAS (Community Modeling and Analysis System) recommends this method, but the scripts included with the AMI are in *bash* rather than in *C-shell* as the former works better with the OS of the AMI. The easiest way to prepare and run a CMAQ case is probably to modify the example included in the AMI. We therefore discuss how to run the 12NE3 2018 benchmarks before turning to a discussion about how to modify the script to accommodate other cases.

Running CMAQ: the 12NE3 2018 benchmark

As previously stated, it is better to run CMAQ using the preloaded scripts. The 12NE3 2018 test case is widely used by the CMAQ community as a benchmark, but it can also serve as a starting point to perform other computations. The AMI comes with the data files necessary to run it; these files are located at /home/ubuntu/CMAQ/data. Running the test case as-it-is simply requires moving into the subdirectory and submitting any of the available scripts:

```
$ cd /home/ubuntu/CMAQ/CCTM/scripts
$ ./CMAQ55_ncores64_12NE3.sh
```

This script will run CMAQ with 64 MPI ranks. The benchmark will run as on any on-premises hardware showing the progress of the simulation. Once the simulation finishes, the output files will be at the /home/ubuntu/CMAQ/data subdirectory unless the script has been modified to route them somewhere else.

CMAQ scripts are usually long and a copy of the one corresponding to run the benchmark is included as appendix A. It is not the purpose here to go over every point of it, because there is already a forum for it, but to discuss a few points to help modify the script to compute other cases. The first section ("Runtime Environment Options") is marked in blue and includes the most basic information about the simulation and environment. The variable CMAQ_HOME should not be modified but CMAQ_DATA needs to point to the subdirectory holding the specific data for that simulation. The variables MECH and APPL described the mechanism ID and name of the simulation, respectively. If you modify the latter, you will also need to change

the variable IMPDIR accordingly. The second section ("CCTM Configuration Options") is marked in dark red and describes some parameters specific to the case; it also controls the granularity of the simulation. Parameters such as dates or time steps need to be adjusted appropriately. The parameters **NPCOL** and **NPROW** dictate the decomposition of the domain and need to be chosen carefully: In addition to determining the size of each subdomain, their product equals the number of MPI ranks that the simulation will run on (in the sample case, that would be 64 ranks). The rest of the section sets up other important variables controlling the simulation. The third section ("Input Directories and Filenames") marked in dark green should only be modified if the inputs for the simulation differ from those in the benchmark. The following sections control the flow of the simulation and should be modified accordingly.

An important question is what to do if you already have your own custom script (that might be used with on-premises hardware) for a specific case. If the script were in *bash*, it would probably only require slight modifications to the subdirectory names (check the Runtime Environment Options section in Appendix A). However, if the script is written in *C-shell*, it is unlikely to run properly and will generate an error message. There are two options: (i) converting the c-shell into a bash one but this requires expertise with both languages; (ii) taking the provided script and changing the values of the parameters to your problem while following the syntax. A third option (only if there is no intellectual property involved in it) is to send us your script and we will provide you with a proper one compatible with the AMI. We recommend not using c-shell scripts as other shells provide a lot more versatility and modern flavor.

CLUSTERS

Over the last 3 years AWS has made great progress in the availability of HPC infrastructure. Even though some of the most recent instances can accommodate large simulations, there is still a need to use clusters to either decrease computational times or when the problem is too large. Something like launching a cluster with thousands or even tens of thousands of cores in the public cloud that a few years seemed impossible or, at least, improbable is now a reality.

AWS has evolved in its own vision of HPC clusters by introducing new tools. For the last few years, the tool of choice (to launch cluster) has been AWS-ParallelCluster. At the end of 2021, the AWS-ParallelCluster Team introduced version 3, which is significantly different from version 3. AWS-ParallelCluster v3 allows model users to launch clusters using different procedures. The two most common ones are: (1) using a virtual environment within an EC2 instance; (2) from a web-based UI. The latter procedure avoids requiring an instance (or other Linux box) with the aid of a web-based UI. However, it has the drawback that the most recent versions of the UI require the ownership of a domain and permissions to perform DNS changes to this domain. We recommend making sure that you have discussed this option with your system/web administrator before proceeding. You can follow the steps described at https://www.pcluster.cloud/01-getting-started.html and contact us (support@odyhpc.com) if you have any questions about this option.

8 – Launching a cluster from a virtual environment

This section assumes that you have completed all the steps described in section 1 including the retrieval of AWS credentials, which is required to launch clusters. As an example, the following discussion assumes the target cluster to be composed of up to 4 c6i.48xlarge instances (for up to 256 cores) or 4 c7i.48xlarge instances (for up to 384 cores). From an instance running Linux (it can be either x86_64 or aarch64), use the following commands[2] to create the virtual environment and install AWS-ParallelCluster and the node version manager:

```
$ mkdir .virtualenv
$ python3 -m pip install --upgrade pip
$ python3 -m pip install --user --upgrade virtualenv
$ python3 -m virtualenv .virtualenv
$ source /home/ubuntu/virtualenv/bin/activate
```

[2] If you are using an OS other than Ubuntu, you will need to modify the root directory accordingly.

```
$ python3 -m pip install --upgrade "aws-parallelcluster"
$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh
| bash
$ chmod ug+x ~/.nvm/nvm.sh
$ source ~/.nvm/nvm.sh
$ nvm install --lts
$ node --version
```

To check the AWS-ParallelCluster version, type

```
$ pcluster version

{
"version": "3.8.0"
}
```

and to install a specific version of AWS-ParallelCluster

```
$ python3 -m pip install "aws-parallelcluster"==3.8.0
```

This sets up the virtual environment, which you need to activate before launching a cluster:

```
$ source /home/ubuntu/virtualenv/bin/activate
$ cd /home/ubuntu/.parallelcluster
$ vi cluster-config.yaml
```

Here, the configuration file is named 'cluster-config.yaml' defining cluster properties. For our first example, the configuration file reads:

```
Region: us-east-1
Image:
 Os: ubuntu2204
 CustomAmi: ami-XXXXXXXXXXXXXXXXX
HeadNode:
 InstanceType: t4g.xlarge
 Networking:
  SubnetId: subnet-XXXXXXXX
 Ssh:
  KeyName: your_own_keypair
 LocalStorage:
  RootVolume:
   Size: 650
```

```
      VolumeType: gp3
    Scheduling:
     Scheduler: slurm
     SlurmQueues:
     - Name: queue1
      CapacityType: SPOT
      ComputeSettings:
       LocalStorage:
        RootVolume:
         Size: 650
         VolumeType: gp3
      ComputeResources:
      - Name: queue1-wrf
       InstanceType: c6i.32xlarge
       Efa:
        Enabled: true
       MinCount: 0
       MaxCount: 4
      Networking:
       SubnetIds:
       - subnet-XXXXXXXX
       PlacementGroup:
        Enabled: true
      Image:
       CustomAmi: ami- XXXXXXXXXXXXXXXXX
```

Launching the cluster requires using the configuration file according to:

```
$ pcluster create-cluster --cluster-configuration cluster-config.yaml
      --cluster-name cluster-WRF256 --region us-east-1
```

This command will spin up the head instance plus all the other components necessary for the cluster. It is possible to check the cluster status with:

```
$ pcluster describe-cluster --cluster-name cluster-WRF256 --region us-east-1
```

Once all the computations have been completed, the cluster and all its components need to be terminated:

```
$ pcluster delete-cluster --region us-east-1 --cluster-name cluster-WRF256
```

To run on either a different number of cores or to use instances belonging to the 7th generation (c7i, m7i, r7i), the above scripts need to be modified accordingly.

9 – Submitting jobs to the cluster

The cluster uses Slurm as the manager. The official website (https://www.schedmd.com/) has every type of information about the scheduler although you only need a fraction of it to run WRF or CMAQ jobs.

Slurm needs a script detailing the specifics of the job such as number of MPI ranks or where the logs are to be written. The AMI includes several examples of these scripts at /home/ubuntu/CLUSTERs to run either WRF or CMAQ jobs. An example of a Slurm script to run WRF reads:

```
#!/bin/bash
#
#SBATCH --job-name=WRF_128
#SBATCH --output=/home/ubuntu/WRF-4.6/logs/out_WRF_128.log
#SBATCH --ntasks=128

cd /home/ubuntu/WRF-4.6/test/em_real
mpirun /home/ubuntu/WRF-4.6/test/em_real/wrf.exe
```

The objective of this script is to run the WRF case placed at /home/ubuntu/WRF-4.6/test/em_real with 128 MPI ranks with the log being written at /home/ubuntu/WRF-4.6/logs. If the script is named 'WRF_128.sbatch,' using the command:

```
$ sbatch WRF_128.sbatch
```

will spin up 4 c6i.32xlarge instances and run the job there. If 2 instances are up and not running any job, this batch file will run the case on the same cluster. If there are no free nodes, either because they are not up or because they are already running some job, the master node will spin up 2 new compute instances.

The scripts to run CMAQ jobs are much more complex as they combine CMAQ scripts with Slurm directives. Appendix B is an example of such script to run the 2018NE benchmark on 256. It is important to remember that the number of MPI ranks defined as #SBATCH –ntasks=256 must match the product of the number of rows times the number of columns.

As usually, if you have any questions with either cluster configuration or the scripts, feel free to contact us (support@odyhpc.com).

Appendix A

Script for running the U.S. Northeast (2018) CMAQv5.4 benchmark

```
#!/bin/bash

# ==================== CCTMv5.4 Run Script ========================
# To report problems or request help with this script
# contact us at support@odyhpc.com
# ================================================================

# ================================================================
#> Runtime Environment Options
# ================================================================

echo 'Start Model Run At ' `date`
export CMAQ_HOME=/home/ubuntu/CMAQ
export CMAQ_DATA=$CMAQ_HOME/data

#> Toggle Diagnostic Mode which will print verbose information to standard output
export CTM_DIAG_LVL=0

#> Choose compiler
export compiler=gcc
export Vrsn=13.2
export compilerString=${compiler}${compilerVrsn}

#> Set General Parameters for Configuring the Simulation
export VRSN=v54           #> Code Version
export PROC=mpi           #> serial or mpi
export MECH=cb6r5_ae7_aq     #> Mechanism ID
export APPL=Bench_2018_12NE3  #> Application Name (e.g. Gridname)

#> Define RUNID as any combination of parameters above or others
export RUNID=${VRSN}_${compilerString}_${APPL}

#> Set the build directory.
export BLD=${CMAQ_HOME}/CCTM/scripts/BLD_CCTM_${VRSN}_${compilerString}
export EXEC=CCTM_${VRSN}.exe

#> Output Each line of Runscript to Log File
if [ $CTM_DIAG_LVL -ne 0 ]
then
  set echo
fi

#> Set Working, Input, and Output Directories
export WORKDIR=${CMAQ_HOME}/CCTM/scripts        #> Working Directory. Where the runscript is.
export OUTDIR=${CMAQ_HOME}/CCTM/output          #> Output Directory
export INPDIR=${CMAQ_DATA}/2018_12NE3           #> Input Directory
export LOGDIR=${OUTDIR}/LOGS                    #> Log Directory Location
export NMLpath=${BLD}                       #> Location of Namelists. Common places are:
                                #>   ${WORKDIR} | ${CCTM_SRC}/MECHS/${MECH} | ${BLD}
export VEXILLA="--mca io_ompi_grouping_option 4 --mca io_ompi_bytes_per_agg 2147483648"
```

```
# ========================================================================
#> CCTM Configuration Options
# ========================================================================
rm -rf $OUTDIR
#> Set Start and End Days for looping
export NEW_START=TRUE           #> Set to FALSE for model restart
export START_DATE="2018-07-01"    #> beginning date (July 1, 2016)
export END_DATE="2018-07-01"      #> ending date    (July 1, 2016)

#> Set Timestepping Parameters
STTIME=000000          #> beginning GMT time (HHMMSS)
NSTEPS=240000          #> time duration (HHMMSS) for this run
TSTEP=010000         #> output time step interval (HHMMSS)

#> Horizontal domain decomposition - assuming MPI
NPCOL=8
NPROW=8
NPROCS=$((NPCOL*NPROW))
export NPCOL_NPROW="$NPCOL $NPROW"

#> Define Execution ID: e.g. [CMAQ-Version-Info]_[User]_[Date]_[Time]
export EXECUTION_ID="CMAQ_CCTM${VRSN}_`id -u -n`_`date -u +%Y%m%d_%H%M%S_%N`"    #> Inform IO/API of
the Execution ID
echo ""
echo "---CMAQ EXECUTION ID: $EXECUTION_ID ---"

#> Keep or Delete Existing Output Files
CLOBBER_DATA=TRUE

#> Logfile Options
#> Master Log File Name; uncomment to write standard output to a log, otherwise write to screen
export LOGFILE=$CMAQ_HOME/$RUNID.log
if [ ! -e $LOGDIR ]
then
  mkdir -p $LOGDIR
fi
export PRINT_PROC_TIME=Y         #> Print timing for all science subprocesses to Logfile
                        #>   [ default: TRUE or Y ]
export STDOUT=T               #> Override I/O-API trying to write information to both the processor
                        #>   logs and STDOUT [ options: T | F ]

export GRID_NAME=2018_12NE3       #> check GRIDDESC file for GRID_NAME options
export GRIDDESC=$INPDIR/GRIDDESC    #> grid description file

#> Retrieve the number of columns, rows, and layers in this simulation
NZ=35
NX=`grep -A 1 ${GRID_NAME} ${GRIDDESC} | tail -1 | sed 's/  */ /g' | cut -d' ' -f6`
NY=`grep -A 1 ${GRID_NAME} ${GRIDDESC} | tail -1 | sed 's/  */ /g' | cut -d' ' -f7`
NCELLS=`echo "${NX} * ${NY} * ${NZ}" | bc -l`

#> Output Species and Layer Options
#> CONC file species; comment or set to "ALL" to write all species to CONC
export CONC_SPCS="O3 NO ANO3I ANO3J NO2 FORM ISOP NH3 ANH4I ANH4J ASO4I ASO4J"
export CONC_BLEV_ELEV=" 1 1" #> CONC file layer range; comment to write all layers to CONC
```

```
#> ACONC file species; comment or set to "ALL" to write all species to ACONC
#export AVG_CONC_SPCS="O3 NO CO NO2 ASO4I ASO4J NH3"
export AVG_CONC_SPCS="ALL"
export ACONC_BLEV_ELEV="1 1" #> ACONC file layer range; comment to write all layers to ACONC
export AVG_FILE_ENDTIME=N    #> override default beginning ACONC timestamp [ default: N ]

#> Synchronization Time Step and Tolerance Options
export CTM_MAXSYNC=300     #> max sync time step (sec) [ default: 720 ]
export CTM_MINSYNC=60      #> min sync time step (sec) [ default: 60 ]
export SIGMA_SYNC_TOP=0.7  #> top sigma level thru which sync step determined [ default: 0.7 ]
#export ADV_HDIV_LIM=0.95  #> maximum horiz. div. limit for adv step adjust [ default: 0.9 ]
export CTM_ADV_CFL=0.95    #> max CFL [ default: 0.75]
#export RB_ATOL=1.0E-09    #> global ROS3 solver absolute tolerance [ default: 1.0E-07 ]

#> Science Options
export CTM_OCEAN_CHEM=Y    #> Flag for ocean halgoen chemistry and sea spray aerosol emissions [ default: Y ]
export CTM_WB_DUST=N       #> use inline windblown dust emissions [ default: Y ]
export CTM_WBDUST_BELD=BELD3 #> landuse database for identifying dust source regions
                  #>   [ default: UNKNOWN ]; ignore if CTM_WB_DUST = N
export CTM_LTNG_NO=N       #> turn on lightning NOx [ default: N ]
export KZMIN=Y             #> use Min Kz option in edyintb [ default: Y ],
                  #>   otherwise revert to Kz0UT
export CTM_MOSAIC=N        #> landuse specific deposition velocities [ default: N ]
export CTM_FST=N           #> mosaic method to get land-use specific stomatal flux
                  #>   [ default: N ]
export PX_VERSION=Y        #> WRF PX LSM
export CLM_VERSION=N       #> WRF CLM LSM
export NOAH_VERSION=N      #> WRF NOAH LSM
export CTM_ABFLUX=Y        #> ammonia bi-directional flux for in-line deposition
                  #>   velocities [ default: N ]
export CTM_BIDI_FERT_NH3=T  #> subtract fertilizer NH3 from emissions because it will be handled
                  #>   by the BiDi calculation [ default: Y ]
export CTM_HGBIDI=N        #> mercury bi-directional flux for in-line deposition
                  #>   velocities [ default: N ]
export CTM_SFC_HONO=Y      #> surface HONO interaction [ default: Y ]
export CTM_GRAV_SETL=Y     #> vdiff aerosol gravitational sedimentation [ default: Y ]
export CTM_BIOGEMIS_BE=Y   #> calculate in-line biogenic emissions with BEIS
export CTM_BIOGEMIS_MG=N   #> turns on MEGAN biogenic emission [ default: N ]
export BDSNP_MEGAN=N       #> turns on BDSNP soil NO emissions [ default: N ]

#> Vertical Extraction Options
export VERTEXT=N
export VERTEXT_COORD_PATH=${WORKDIR}/lonlat.csv

#> I/O Controls
export IOAPI_LOG_WRITE=F   #> turn on excess WRITE3 logging [ options: T | F ]
export FL_ERR_STOP=N       #> stop on inconsistent input files
export PROMPTFLAG=F        #> turn on I/O-API PROMPT*FILE interactive mode [ options: T | F ]
export IOAPI_OFFSET_64=YES  #> support large timestep records (>2GB/timestep record) [ options: YES | NO ]
export IOAPI_CHECK_HEADERS=N #> check file headers [ options: Y | N ]
export CTM_EMISCHK=N       #> Abort CMAQ if missing surrogates from emissions Input files
export EMISDIAG=F          #> Print Emission Rates at the output time step after they have been
                  #>  scaled and modified by the user Rules [options: F | T or 2D | 3D | 2DSUM ]
                  #>  Individual streams can be modified using the variables:
                  #>    GR_EMIS_DIAG_## | STK_EMIS_DIAG_## | BIOG_EMIS_DIAG
```

```
                #>     MG_EMIS_DIAG   | LTNG_EMIS_DIAG   | DUST_EMIS_DIAG
                #>     SEASPRAY_EMIS_DIAG
                #>  Note that these diagnostics are different than other emissions diagnostic
                #>  output because they occur after scaling.
export EMISDIAG_SUM=F      #> Print Sum of Emission Rates to Gridded Diagnostic File

#> Diagnostic Output Flags
export CTM_CKSUM=Y        #> checksum report [ default: Y ]
export CLD_DIAG=N         #> cloud diagnostic file [ default: N ]

export CTM_PHOTDIAG=N      #> photolysis diagnostic file [ default: N ]
export NLAYS_PHOTDIAG="1"   #> Number of layers for PHOTDIAG2 and PHOTDIAG3 from
                #>    Layer 1 to NLAYS_PHOTDIAG  [ default: all layers ]
#export NWAVE_PHOTDIAG="294 303 310 316 333 381 607"  #> Wavelengths written for variables
                             #>  in PHOTDIAG2 and PHOTDIAG3
                             #>  [ default: all wavelengths ]

export CTM_SSEMDIAG=N      #> sea-spray emissions diagnostic file [ default: N ]
export CTM_DUSTEM_DIAG=N    #> windblown dust emissions diagnostic file [ default: N ];
                #>    Ignore if CTM_WB_DUST = N
export CTM_DEPV_FILE=N     #> deposition velocities diagnostic file [ default: N ]
export VDIFF_DIAG_FILE=N    #> vdiff & possibly aero grav. sedimentation diagnostic file [ default: N ]
export LTNGDIAG=N         #> lightning diagnostic file [ default: N ]
export B3GTS_DIAG=N        #> BEIS mass emissions diagnostic file [ default: N ]
export CTM_WVEL=Y         #> save derived vertical velocity component to conc
                #>    file [ default: Y ]

# ========================================================================
#> Input Directories and Filenames
# ========================================================================

ICpath=$INPDIR/icbc                  #> initial conditions input directory
BCpath=$INPDIR/icbc                  #> boundary conditions input directory
EMISpath=$INPDIR/emis                #> gridded emissions input directory
IN_PTpath=$INPDIR/emis               #> point source emissions input directory
IN_LTpath=$INPDIR/lightning          #> lightning NOx input directory
METpath=$INPDIR/met/mcipv5.4         #> meteorology input directory
#JVALpath=$INPDIR/jproc              #> offline photolysis rate table directory
OMIpath=$BLD                         #> ozone column data for the photolysis model
EPICpath=$INPDIR/epic                #> BELD landuse data for windblown dust model
SZpath=$INPDIR/surface               #> surf zone file for in-line seaspray emissions

# ========================================================================
#> Begin Loop Through Simulation Days
# ========================================================================
rtarray=""

TODAYG=${START_DATE}
TODAYJ=`date -ud "${START_DATE}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ
START_DAY=${TODAYJ}
STOP_DAY=`date -ud "${END_DATE}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ
NDAYS=0

while [ $TODAYJ -le $STOP_DAY ]          #>Compare dates in terms of YYYYJJJ
do
```

```
 NDAYS=`echo "${NDAYS} + 1" | bc -l`

 #> Retrieve Calendar day Information
 YYYYMMDD=`date -ud "${TODAYG}" +%Y%m%d` #> Convert YYYY-MM-DD to YYYYMMDD
 YYYYMM=`date -ud "${TODAYG}" +%Y%m`     #> Convert YYYY-MM-DD to YYYYMM
 YYMMDD=`date -ud "${TODAYG}" +%y%m%d`   #> Convert YYYY-MM-DD to YYMMDD
 YYYYJJJ=$TODAYJ

 #> Calculate Yesterday's Date
 YESTERDAY=`date -ud "${TODAYG}-1days" +%Y%m%d` #> Convert YYYY-MM-DD to YYYYJJJ
# ========================================================================
#> Set Output String and Propagate Model Configuration Documentation
# ========================================================================
 echo ""
 echo "Set up input and output files for Day ${TODAYG}."

 #> set output file name extensions
 export CTM_APPL=${RUNID}_${YYYYMMDD}

 #> Copy Model Configuration To Output Folder
 if [ ! -d $OUTDIR ]; then
   mkdir -p $OUTDIR
 fi
 cp $BLD/CCTM_${VRSN}.cfg $OUTDIR/CCTM_${CTM_APPL}.cfg

# ========================================================================
#> Input Files (Some are Day-Dependent)
# ========================================================================

 #> Initial conditions
 if [ "$NEW_START" == "true" ] || [ "$NEW_START" == "TRUE" ]
 then
   export ICFILE=CCTM_ICON_v54_${MECH}_12NE3_20180701.nc
   export INIT_MEDC_1=notused
   export INITIAL_RUN=Y #related to restart soil information file
 else
   ICpath=$OUTDIR
   export ICFILE=CCTM_CGRID_${RUNID}_${YESTERDAY}.nc
   export INIT_MEDC_1=$ICpath/CCTM_MEDIA_CONC_${RUNID}_${YESTERDAY}.nc
   export INITIAL_RUN=N
 fi

 #> Boundary conditions
 BCFILE=CCTM_BCON_v54_${MECH}_12NE3_${YYYYMMDD}.nc

 echo "I/BCFILE $ICFILE $BCFILE"
 #> Off-line photolysis rates
 #export JVALfile=JTABLE_${YYYYJJJ}

 #> Ozone column data
 OMIfile=OMI_1979_to_2019.dat

 #> Optics file
 OPTfile=PHOT_OPTICS.dat
```

```
 #> MCIP meteorology files
 export GRID_BDY_2D=$METpath/GRIDBDY2D_12NE3_${YYYYMMDD}.nc  # GRID files are static, not day-specific
 export GRID_CRO_2D=$METpath/GRIDCRO2D_12NE3_${YYYYMMDD}.nc
 export GRID_CRO_3D=$METpath/GRIDCRO3D_12NE3_${YYYYMMDD}.nc
 export GRID_DOT_2D=$METpath/GRIDDOT2D_12NE3_${YYYYMMDD}.nc
 export MET_CRO_2D=$METpath/METCRO2D_12NE3_${YYYYMMDD}.nc
 export MET_CRO_3D=$METpath/METCRO3D_12NE3_${YYYYMMDD}.nc
 export MET_DOT_3D=$METpath/METDOT3D_12NE3_${YYYYMMDD}.nc
 export MET_BDY_3D=$METpath/METBDY3D_12NE3_${YYYYMMDD}.nc
 export LUFRAC_CRO=$METpath/LUFRAC_CRO_12NE3_${YYYYMMDD}.nc

 #> Control Files
 #>
 #> IMPORTANT NOTE
 #>
 #> The DESID control files defined below are an integral part of controlling the behavior of the model simulation.
 #> Among other things, they control the mapping of species in the emission files to chemical species in the model
and
 #> several aspects related to the simulation of organic aerosols.
 #> Please carefully review the DESID control files to ensure that they are configured to be consistent with the
assumptions
 #> made when creating the emission files defined below and the desired representation of organic aerosols.
 #> For further information, please see:
 #> + AERO7 Release Notes section on 'Required emission updates':
 #>   https://github.com/USEPA/CMAQ/blob/master/DOCS/Release_Notes/aero7_overview.md
 #> + CMAQ User's Guide section 6.9.3 on 'Emission Compatability':
 #>
https://github.com/USEPA/CMAQ/blob/master/DOCS/Users_Guide/CMAQ_UG_ch06_model_configuration_option
s.md#6.9.3_Emission_Compatability
 #> + Emission Control (DESID) Documentation in the CMAQ User's Guide:
 #>
https://github.com/USEPA/CMAQ/blob/master/DOCS/Users_Guide/Appendix/CMAQ_UG_appendixB_emissions_c
ontrol.md
 #>
 export DESID_CTRL_NML=${BLD}/CMAQ_Control_DESID.nml
 export DESID_CHEM_CTRL_NML=${BLD}/CMAQ_Control_DESID_${MECH}.nml

 #> The following namelist configures aggregated output (via the Explicit and Lumped
 #> Air Quality Model Output (ELMO) Module), domain-wide budget output, and chemical
 #> family output.
 export MISC_CTRL_NML=${BLD}/CMAQ_Control_Misc.nml

 #> The following namelist controls the mapping of meteorological land use types and the NH3 and Hg emission
 #> potentials
 export STAGECTRL_NML=${BLD}/CMAQ_Control_STAGE.nml

 #> Spatial Masks For Emissions Scaling
 #export CMAQ_MASKS=$SZpath/OCEAN_${MM}_L3m_MC_CHL_chlor_a_12NE3.nc #> horizontal grid-dependent
ocean file
 export CMAQ_MASKS=$INPDIR/GRIDMASK_STATES_12NE3.nc

 #> Gridded Emissions Files
 export N_EMIS_GR=2
 EMISfile=emis_mole_all_${YYYYMMDD}_12NE3_nobeis_norwc_2018gc_cb6_18j.ncf
 export GR_EMIS_001=${EMISpath}/merged_nobeis_norwc/${EMISfile}
```

```
export GR_EMIS_LAB_001=GRIDDED_EMIS
export GR_EM_SYM_DATE_001=F # To change default behaviour please see Users Guide for EMIS_SYM_DATE

EMISfile=emis_mole_rwc_${YYYYMMDD}_12NE3_cmaq_cb6ae7_2018gc_cb6_18j.ncf
export GR_EMIS_002=${EMISpath}/rwc/${EMISfile}
export GR_EMIS_LAB_002=GR_RES_FIRES
export GR_EM_SYM_DATE_002=F # To change default behaviour please see Users Guide for EMIS_SYM_DATE

#> In-line point emissions configuration
export N_EMIS_PT=10        #> Number of elevated source groups

STKCASEG=12US1_2018gc_cb6_18j          # Stack Group Version Label
STKCASEE=12US1_cmaq_cb6ae7_2018gc_cb6_18j  # Stack Emission Version Label

# Time-Independent Stack Parameters for Inline Point Sources
export STK_GRPS_001=$IN_PTpath/ptnonipm/stack_groups_ptnonipm_${STKCASEG}.ncf
export STK_GRPS_002=$IN_PTpath/ptegu/stack_groups_ptegu_${STKCASEG}.ncf
export STK_GRPS_003=$IN_PTpath/othpt/stack_groups_othpt_${STKCASEG}.ncf
export STK_GRPS_004=$IN_PTpath/ptagfire/stack_groups_ptagfire_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_005=$IN_PTpath/ptfire-rx/stack_groups_ptfire-rx_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_006=$IN_PTpath/ptfire-wild/stack_groups_ptfire-wild_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_007=$IN_PTpath/ptfire_othna/stack_groups_ptfire_othna_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_008=$IN_PTpath/pt_oilgas/stack_groups_pt_oilgas_${STKCASEG}.ncf
export STK_GRPS_009=$IN_PTpath/cmv_c3_12/stack_groups_cmv_c3_12_${STKCASEG}.ncf
export STK_GRPS_010=$IN_PTpath/cmv_c1c2_12/stack_groups_cmv_c1c2_12_${STKCASEG}.ncf

# Emission Rates for Inline Point Sources
export STK_EMIS_001=$IN_PTpath/ptnonipm/inln_mole_ptnonipm_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_002=$IN_PTpath/ptegu/inln_mole_ptegu_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_003=$IN_PTpath/othpt/inln_mole_othpt_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_004=$IN_PTpath/ptagfire/inln_mole_ptagfire_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_005=$IN_PTpath/ptfire-rx/inln_mole_ptfire-rx_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_006=$IN_PTpath/ptfire-wild/inln_mole_ptfire-wild_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_007=$IN_PTpath/ptfire_othna/inln_mole_ptfire_othna_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_008=$IN_PTpath/pt_oilgas/inln_mole_pt_oilgas_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_009=$IN_PTpath/cmv_c3_12/inln_mole_cmv_c3_12_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_010=$IN_PTpath/cmv_c1c2_12/inln_mole_cmv_c1c2_12_${YYYYMMDD}_${STKCASEE}.ncf

# Label Each Emissions Stream
export STK_EMIS_LAB_001=PT_NONEGU
export STK_EMIS_LAB_002=PT_EGU
export STK_EMIS_LAB_003=PT_OTHER
export STK_EMIS_LAB_004=PT_AGFIRES
export STK_EMIS_LAB_005=PT_RXFIRES
export STK_EMIS_LAB_006=PT_WILDFIRES
export STK_EMIS_LAB_007=PT_OTHFIRES
export STK_EMIS_LAB_008=PT_OILGAS
export STK_EMIS_LAB_009=PT_CMV_C3
export STK_EMIS_LAB_010=PT_CMV_C1C2

# Allow CMAQ to Use Point Source files with dates that do not
# match the internal model date
# To change default behaviour please see Users Guide for EMIS_SYM_DATE
export STK_EM_SYM_DATE_001=F
export STK_EM_SYM_DATE_002=F
```

```
export STK_EM_SYM_DATE_003=F
export STK_EM_SYM_DATE_004=F
export STK_EM_SYM_DATE_005=F
export STK_EM_SYM_DATE_006=F
export STK_EM_SYM_DATE_007=F
export STK_EM_SYM_DATE_008=F

#> Lightning NOx configuration
if [ "$CTM_LTNG_NO" == "Y" ]
then
  export LTNGNO="InLine"    #> set LTNGNO to "Inline" to activate in-line calculation

#> In-line lightning NOx options
  export USE_NLDN=Y       #> use hourly NLDN strike file [ default: Y ]
  if [ "$USE_NLDN" == "Y" ]
  then
    export NLDN_STRIKES=${IN_LTpath}/NLDN_12km_60min_${YYYYMMDD}.ioapi
  fi
  export LTNGPARMS_FILE=${IN_LTpath}/LTNG_AllParms_12NE3.nc #> lightning parameter file
fi

#> In-line biogenic emissions configuration
if [ "$CTM_BIOGEMIS_BE" == "Y" ]
then
  IN_BEISpath=${INPDIR}/surface
  export GSPRO=$BLD/gspro_biogenics.txt
  export BEIS_NORM_EMIS=$IN_BEISpath/beis4_beld6_norm_emis.12NE3.nc
  export BEIS_SOILINP=$OUTDIR/CCTM_BSOILOUT_${RUNID}_${YESTERDAY}.nc
                #> Biogenic NO soil input file; ignore if NEW_START = TRUE
fi
if [ "$CTM_BIOGEMIS_MG" == "Y" ]
then
 export MEGAN_SOILINP=$OUTDIR/CCTM_MSOILOUT_${RUNID}_${YESTERDAY}.nc
                #> Biogenic NO soil input file; ignore if INITIAL_RUN = Y
                #>                    ; ignore if IGNORE_SOILINP = Y
    export MEGAN_CTS=$SZpath/megan3.2/CT3_CONUS.ncf
    export MEGAN_EFS=$SZpath/megan3.2/EFMAPS_CONUS.ncf
    export MEGAN_LDF=$SZpath/megan3.2/LDF_CONUS.ncf
    if [ "$BDSNP_MEGAN" == "Y"]
    then
      export BDSNPINP=$OUTDIR/CCTM_BDSNPOUT_${RUNID}_${YESTERDAY}.nc
      export BDSNP_FFILE=$SZpath/megan3.2/FERT_tceq_12km.ncf
      export BDSNP_NFILE=$SZpath/megan3.2/NDEP_tceq_12km.ncf
      export BDSNP_LFILE=$SZpath/megan3.2/LANDTYPE_tceq_12km.ncf
      export BDSNP_AFILE=$SZpath/megan3.2/ARID_tceq_12km.ncf
      export BDSNP_NAFILE=$SZpath/megan3.2/NONARID_tceq_12km.ncf
    fi
fi

#> In-line sea spray emissions configuration
export OCEAN_1=$SZpath/OCEAN_${MM}_L3m_MC_CHL_chlor_a_12NE3.nc #> horizontal grid-dependent ocean
file

#> Bidirectional ammonia configuration
if [ "$CTM_ABFLUX" == "Y" ]
```

```
 then
   export E2C_SOIL=${EPICpath}/2018r1_EPIC0509_12NE3_soil.nc
   export E2C_CHEM=${EPICpath}/2018r1_EPIC0509_12NE3_time${YYYYMMDD}.nc
   export E2C_CHEM_YEST=${EPICpath}/2018r1_EPIC0509_12NE3_time${YESTERDAY}.nc
   export E2C_LU=${EPICpath}/beld4_12NE3_2011.nc
 fi

#> Inline Process Analysis
 export CTM_PROCAN=N       #> use process analysis [ default: N]
#  if ( $?CTM_PROCAN ) then   # $CTM_PROCAN is defined
#    if ( $CTM_PROCAN == 'Y' || $CTM_PROCAN == 'T' ) then
##> process analysis global column, row and layer ranges
##      setenv PA_BCOL_ECOL "10 90"  # default: all columns
##      setenv PA_BROW_EROW "10 80"  # default: all rows
##      setenv PA_BLEV_ELEV "1  4"   # default: all levels
#      setenv PACM_INFILE ${NMLpath}/pa_${MECH}.ctl
#      setenv PACM_REPORT $OUTDIR/"PA_REPORT".${YYYYMMDD}
#    endif
#  endif

#> Sulfur Tracking Model (STM)
 export STM_SO4TRACK=N       #> sulfur tracking [ default: N ]
# if ( $?STM_SO4TRACK ) then
#    if ( $STM_SO4TRACK == 'Y' || $STM_SO4TRACK == 'T' ) then

#     #> option to normalize sulfate tracers [ default: Y ]
#     setenv STM_ADJSO4 Y

#    endif
# endif

# ========================================================================
#> Output Files
# ========================================================================

 #> set output file names
 export S_CGRID="$OUTDIR/CCTM_CGRID_${CTM_APPL}.nc"              #> 3D Inst. Concentrations
 export CTM_CONC_1="$OUTDIR/CCTM_CONC_${CTM_APPL}.nc -v"          #> On-Hour Concentrations
 export A_CONC_1="$OUTDIR/CCTM_ACONC_${CTM_APPL}.nc -v"           #> Hourly Avg. Concentrations
 export MEDIA_CONC="$OUTDIR/CCTM_MEDIA_CONC_${CTM_APPL}.nc -v"      #> NH3 Conc. in Media
 export CTM_DRY_DEP_1="$OUTDIR/CCTM_DRYDEP_${CTM_APPL}.nc -v"       #> Hourly Dry Deposition
 export CTM_DEPV_DIAG="$OUTDIR/CCTM_DEPV_${CTM_APPL}.nc -v"        #> Dry Deposition Velocities
 export B3GTS_S="$OUTDIR/CCTM_B3GTS_S_${CTM_APPL}.nc -v"          #> Biogenic Emissions
 export SOILOUT="$OUTDIR/CCTM_SOILOUT_${CTM_APPL}.nc"             #> Soil Emissions
 export CTM_WET_DEP_1="$OUTDIR/CCTM_WETDEP1_${CTM_APPL}.nc -v"      #> Wet Dep From All Clouds
 export CTM_WET_DEP_2="$OUTDIR/CCTM_WETDEP2_${CTM_APPL}.nc -v"      #> Wet Dep From SubGrid Clouds
 export CTM_PMDIAG_1="$OUTDIR/CCTM_PMDIAG_${CTM_APPL}.nc -v"       #> On-Hour Particle Diagnostics
 export CTM_APMDIAG_1="$OUTDIR/CCTM_APMDIAG_${CTM_APPL}.nc -v"      #> Hourly Avg. Particle Diagnostics
 export CTM_RJ_1="$OUTDIR/CCTM_PHOTDIAG1_${CTM_APPL}.nc -v"        #> 2D Surface Summary from Inline
Photolysis
 export CTM_RJ_2="$OUTDIR/CCTM_PHOTDIAG2_${CTM_APPL}.nc -v"        #> 3D Photolysis Rates
 export CTM_RJ_3="$OUTDIR/CCTM_PHOTDIAG3_${CTM_APPL}.nc -v"        #> 3D Optical and Radiative Results from
Photolysis
 export CTM_SSEMIS_1="$OUTDIR/CCTM_SSEMIS_${CTM_APPL}.nc -v"       #> Sea Spray Emissions
 export CTM_DUST_EMIS_1="$OUTDIR/CCTM_DUSTEMIS_${CTM_APPL}.nc -v"    #> Dust Emissions
```

```
 export CTM_IPR_1="$OUTDIR/CCTM_PA_1_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IPR_2="$OUTDIR/CCTM_PA_2_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IPR_3="$OUTDIR/CCTM_PA_3_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IRR_1="$OUTDIR/CCTM_IRR_1_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_IRR_2="$OUTDIR/CCTM_IRR_2_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_IRR_3="$OUTDIR/CCTM_IRR_3_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_DRY_DEP_MOS="$OUTDIR/CCTM_DDMOS_${CTM_APPL}.nc -v"     #> Dry Dep
 export CTM_DRY_DEP_FST="$OUTDIR/CCTM_DDFST_${CTM_APPL}.nc -v"     #> Dry Dep
 export CTM_DEPV_MOS="$OUTDIR/CCTM_DEPVMOS_${CTM_APPL}.nc -v"      #> Dry Dep Velocity
 export CTM_DEPV_FST="$OUTDIR/CCTM_DEPVFST_${CTM_APPL}.nc -v"      #> Dry Dep Velocity
 export CTM_VDIFF_DIAG="$OUTDIR/CCTM_VDIFF_DIAG_${CTM_APPL}.nc -v"  #> Vertical Dispersion Diagnostic
 export CTM_VSED_DIAG="$OUTDIR/CCTM_VSED_DIAG_${CTM_APPL}.nc -v"    #> Particle Grav. Settling Velocity
 export CTM_LTNGDIAG_1="$OUTDIR/CCTM_LTNGHRLY_${CTM_APPL}.nc -v"    #> Hourly Avg Lightning NO
 export CTM_LTNGDIAG_2="$OUTDIR/CCTM_LTNGCOL_${CTM_APPL}.nc -v"     #> Column Total Lightning NO
 export CTM_VEXT_1="$OUTDIR/CCTM_VEXT_${CTM_APPL}.nc -v"          #> On-Hour 3D Concs at select sites

 #> set floor file (neg concs)
 export FLOOR_FILE=${OUTDIR}/FLOOR_${CTM_APPL}.txt

 #> look for existing log files and output files
 ( ls CTM_LOG_???.${CTM_APPL} > buff.txt ) >& /dev/null
 ( ls ${LOGDIR}/CTM_LOG_???.${CTM_APPL} >> buff.txt ) >& /dev/null
 log_test=`cat buff.txt`; rm -f buff.txt

 OUT_FILES=(${FLOOR_FILE} ${S_CGRID} ${CTM_CONC_1} ${A_CONC_1} ${MEDIA_CONC}              \
        ${CTM_DRY_DEP_1} $CTM_DEPV_DIAG $B3GTS_S $SOILOUT $CTM_WET_DEP_1              \
        $CTM_WET_DEP_2 $CTM_PMDIAG_1 $CTM_APMDIAG_1                                   \
        $CTM_RJ_1 $CTM_RJ_2 $CTM_RJ_3 $CTM_SSEMIS_1 $CTM_DUST_EMIS_1 $CTM_IPR_1 $CTM_IPR_2  \
        $CTM_IPR_3 $CTM_IRR_1 $CTM_IRR_2 $CTM_IRR_3 $CTM_DRY_DEP_MOS                  \
        $CTM_DRY_DEP_FST $CTM_DEPV_MOS $CTM_DEPV_FST $CTM_VDIFF_DIAG $CTM_VSED_DIAG       \
        $CTM_LTNGDIAG_1 $CTM_LTNGDIAG_2 $CTM_VEXT_1 )

 OUT_FILES=`echo $OUT_FILES | sed "s; -v;;g" | sed "s;MPI:;;g" `
 ( ls $OUT_FILES > buff.txt ) >& /dev/null
 out_test=`cat buff.txt`; rm -f buff.txt

 #> delete previous output if requested
 if [ "$CLOBBER_DATA" == "true" ] || [ "$CLOBBER_DATA" == "TRUE" ]
 then
   echo
   echo "Existing Logs and Output Files for Day ${TODAYG} Will Be Deleted"

   #> remove previous log files
   for file in $log_test ; do
#      #echo "Deleting log file: $file"
     /bin/rm -f "$file"
   done
   #> remove previous output files
   for file in $out_test ; do
#      #echo "Deleting output file: $file"
     /bin/rm -f "$file"
   done
   /bin/rm -f ${OUTDIR}/CCTM_EMDIAG*${RUNID}_${YYYYMMDD}.nc

 else
```

```
  #> error if previous log files exist
  if [ "$log_test" -ne "" ]
  then
    echo "*** Logs exist - run ABORTED ***"
    echo "*** To overide, set CLOBBER_DATA = TRUE in run_cctm.csh ***"
    echo "*** and these files will be automatically deleted. ***"
    exit 1
 fi

  #> error if previous output files exist
  if [ "$out_test" != "" ]
  then
    echo "*** Output Files Exist - run will be ABORTED ***"
   for file in $out_test ; do
     echo " cannot delete $file"
   done
    echo "*** To overide, set CLOBBER_DATA = TRUE in run_cctm.csh ***"
    echo "*** and these files will be automatically deleted. ***"
    exit 1
   fi
fi

 #> for the run control ...
 export CTM_STDATE=$YYYYJJJ
 export CTM_STTIME=$STTIME
 export CTM_RUNLEN=$NSTEPS
 export CTM_TSTEP=$TSTEP
 export INIT_CONC_1=$ICpath/$ICFILE
 export BNDY_CONC_1=$BCpath/$BCFILE
 export OMI=$OMIpath/$OMIfile
 export OPTICS_DATA=$OMIpath/$OPTfile
#export XJ_DATA=$JVALpath/$JVALfile

 #> species defn & photolysis
 export gc_matrix_nml=${NMLpath}/GC_$MECH.nml
 export ae_matrix_nml=${NMLpath}/AE_$MECH.nml
 export nr_matrix_nml=${NMLpath}/NR_$MECH.nml
 export tr_matrix_nml=${NMLpath}/Species_Table_TR_0.nml

 #> check for photolysis input data
 export CSQY_DATA=${NMLpath}/CSQY_DATA_$MECH


#|#  if [ ! (-e $CSQY_DATA ) ]
#|#  then
#|#    echo " $CSQY_DATA  not found "
#|#    exit 1
#|#  fi
#|#  if [ ! (-e $OPTICS_DATA ) ]
#|#  then
#|#    echo " $OPTICS_DATA  not found "
#|#    exit 1
#|#  fi

# ======================================================================
```

```
#> Execution Portion
# ========================================================================

 #> Print Startup Dialogue Information to Standard Out
 echo
 echo "CMAQ Processing of Day $YYYYMMDD Began at `date`"
 echo

 #> Executable call for single PE, uncomment to invoke
 #( /usr/bin/time -p $BLD/$EXEC ) |& tee buff_${EXECUTION_ID}.txt

 #> Executable call for multi PE, configure for your system
 # MPI=/usr/local/intel/impi/3.2.2.006/bin64
 # MPIRUN=$MPI/mpirun
 ( /usr/bin/time -p mpirun -np $NPROCS $VEXILLA $BLD/$EXEC ) |& tee buff_${EXECUTION_ID}.txt

 #> Harvest Timing Output so that it may be reported below
 rtarray="${rtarray} `tail -3 buff_${EXECUTION_ID}.txt | grep -Eo '[+-]?[0-9]+([.][0-9]+)?' | head -1` "
 rm -rf buff_${EXECUTION_ID}.txt

 #> Abort script if abnormal termination
 if [ ! -e $OUTDIR/CCTM_CGRID_${CTM_APPL}.nc ]
 then
  echo ""
  echo "**************************************************************"
  echo "** Runscript Detected an Error: CGRID file was not written. **"
  echo "**   This indicates that CMAQ was interrupted or an issue   **"
  echo "**   exists with writing output. The runscript will now     **"
  echo "**   abort rather than proceeding to subsequent days.       **"
  echo "**************************************************************"
  break
 fi

 #> Print Concluding Text
 echo
 echo "CMAQ Processing of Day $YYYYMMDD Finished at `date`"
 echo
 echo "\\\\\=====\\\\\=====\\\\\=====\\\\\=====/////=====/////=====/////=====/////"
 echo

# ========================================================================
#> Finalize Run for This Day and Loop to Next Day
# ========================================================================

 #> Save Log Files and Move on to Next Simulation Day
 echo "CTM_LOG_???.${CTM_APPL} and LOGDIR $LOGDIR"
 mv CTM_LOG_???.${CTM_APPL} $LOGDIR
 if [ $CTM_DIAG_LVL != 0 ]
 then
  mv CTM_DIAG_???.${CTM_APPL} $LOGDIR
 fi

 #> The next simulation day will, by definition, be a restart
 export NEW_START=false
```

```
 #> Increment both Gregorian and Julian Days
 TODAYG=`date -ud "${TODAYG}+1days" +%Y-%m-%d` #> Add a day for tomorrow
 TODAYJ=`date -ud "${TODAYG}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ

done
#|#done  #> Loop to the next Simulation Day

# ======================================================================
#> Generate Timing Report
# ======================================================================
RTMTOT=0
for it in `seq ${NDAYS}` ; do
   rt=`echo ${rtarray} | cut -d' ' -f${it}`
   RTMTOT=`echo "${RTMTOT} + ${rt}" | bc -l`
done

RTMAVG=`echo "scale=2; ${RTMTOT} / ${NDAYS}" | bc -l`
RTMTOT=`echo "scale=2; ${RTMTOT} / 1" | bc -l`

echo
echo "================================="
echo "  ***** CMAQ TIMING REPORT *****"
echo "================================="
echo "Start Day: ${START_DATE}"
echo "End Day:   ${END_DATE}"
echo "Number of Simulation Days: ${NDAYS}"
echo "Domain Name:              ${GRID_NAME}"
echo "Number of Grid Cells:     ${NCELLS}  (ROW x COL x LAY)"
echo "Number of Layers:         ${NZ}"
echo "Number of Processes:      ${NPROCS}"
echo "   All times are in seconds."
echo
echo "Num  Day       Wall Time"
d=0
day=${START_DATE}

for it in `seq ${NDAYS}` ; do
   # Set the right day and format it
   d=`echo "${d} + 1"  | bc -l`
   n=`printf "%02d" ${d}`

   # Choose the correct time variables
   rt=`echo ${rtarray} | cut -d' ' -f${it}`

   # Write out row of timing data
   echo "${n}   ${day}   ${rt}"

   # Increment day for next loop
   day=`date -ud "${day}+1days" +%Y-%m-%d`
done

echo "    Total Time = ${RTMTOT}"
echo "     Avg. Time = ${RTMAVG}"

exit
```

Appendix B

Script for running the U.S. Northeast (2018) benchmark in a cluster

```
#!/bin/bash
#
#SBATCH --job-name=CMAQ_cluster_256                                    # Job-name = CMAQ_cluster_256
#SBATCH --output=/home/ubuntu/CMAQ/CCTM/output/out_CMAQ256.log         # Standard output

#SBATCH --ntasks=256                                                   # Number of MPI ranks
cd /home/ubuntu/CMAQ/CCTM/scripts

# ========================================================================
#               CCTMv5.4 cluster Script
#     To report problems or request help with this script
#           contact us at support@odyhpc.com
# ========================================================================

# ========================================================================
#> Runtime Environment Options
# ========================================================================

echo 'Start Model Run At ' `date`
export CMAQ_HOME=/home/ubuntu/CMAQ
export CMAQ_DATA=$CMAQ_HOME/data

#> Toggle Diagnostic Mode which will print verbose information to standard output
export CTM_DIAG_LVL=0

#> Choose compiler
export compiler=gcc
export Vrsn=13.2
export compilerString=${compiler}${compilerVrsn}

#> Set General Parameters for Configuring the Simulation
export VRSN=v54          #> Code Version
export PROC=mpi          #> serial or mpi
export MECH=cb6r5_ae7_aq     #> Mechanism ID
export APPL=Bench_2018_12NE3  #> Application Name (e.g. Gridname)

#> Define RUNID as any combination of parameters above or others
export RUNID=${VRSN}_${compilerString}_${APPL}

#> Set the build directory.
export BLD=${CMAQ_HOME}/CCTM/scripts/BLD_CCTM_${VRSN}_${compilerString}
export EXEC=CCTM_${VRSN}.exe

#> Output Each line of Runscript to Log File
if [ $CTM_DIAG_LVL -ne 0 ]
then
  set echo
fi

#> Set Working, Input, and Output Directories
export WORKDIR=${CMAQ_HOME}/CCTM/scripts       #> Working Directory. Where the runscript is.
export OUTDIR=${CMAQ_HOME}/CCTM/output         #> Output Directory
```

```
export INPDIR=${CMAQ_DATA}/2018_12NE3          #> Input Directory
export LOGDIR=${OUTDIR}/LOGS                   #> Log Directory Location
export NMLpath=${BLD}                          #> Location of Namelists. Common places are:
                                  #>   ${WORKDIR} | ${CCTM_SRC}/MECHS/${MECH} | ${BLD}
export VEXILLA="--mca io_ompi_grouping_option 4 --mca io_ompi_bytes_per_agg 2147483648"
# ======================================================================
#> CCTM Configuration Options
# ======================================================================
rm -rf $OUTDIR
#> Set Start and End Days for looping
export NEW_START=TRUE           #> Set to FALSE for model restart
export START_DATE="2018-07-01"    #> beginning date (July 1, 2016)
export END_DATE="2018-07-01"      #> ending date    (July 1, 2016)

#> Set Timestepping Parameters
STTIME=000000          #> beginning GMT time (HHMMSS)
NSTEPS=240000          #> time duration (HHMMSS) for this run
TSTEP=010000          #> output time step interval (HHMMSS)

#> Horizontal domain decomposition - assuming MPI
NPCOL=16
NPROW=16
NPROCS=$((NPCOL*NPROW))
export NPCOL_NPROW="$NPCOL $NPROW"

#> Define Execution ID: e.g. [CMAQ-Version-Info]_[User]_[Date]_[Time]
export EXECUTION_ID="CMAQ_CCTM${VRSN}_`id -u -n`_`date -u +%Y%m%d_%H%M%S_%N`"    #> Inform IO/API of
the Execution ID
echo ""
echo "---CMAQ EXECUTION ID: $EXECUTION_ID ---"

#> Keep or Delete Existing Output Files
CLOBBER_DATA=TRUE

#> Logfile Options
#> Master Log File Name; uncomment to write standard output to a log, otherwise write to screen
export LOGFILE=$CMAQ_HOME/$RUNID.log
if [ ! -e $LOGDIR ]
then
  mkdir -p $LOGDIR
fi
export PRINT_PROC_TIME=Y         #> Print timing for all science subprocesses to Logfile
                         #>   [ default: TRUE or Y ]
export STDOUT=T                  #> Override I/O-API trying to write information to both the processor
                         #>   logs and STDOUT [ options: T | F ]

export GRID_NAME=2018_12NE3       #> check GRIDDESC file for GRID_NAME options
export GRIDDESC=$INPDIR/GRIDDESC    #> grid description file

#> Retrieve the number of columns, rows, and layers in this simulation
NZ=35
NX=`grep -A 1 ${GRID_NAME} ${GRIDDESC} | tail -1 | sed 's/  */ /g' | cut -d' ' -f6`
NY=`grep -A 1 ${GRID_NAME} ${GRIDDESC} | tail -1 | sed 's/  */ /g' | cut -d' ' -f7`
NCELLS=`echo "${NX} * ${NY} * ${NZ}" | bc -l`
```

```
#> Output Species and Layer Options
#> CONC file species; comment or set to "ALL" to write all species to CONC
export CONC_SPCS="O3 NO ANO3I ANO3J NO2 FORM ISOP NH3 ANH4I ANH4J ASO4I ASO4J"
export CONC_BLEV_ELEV=" 1 1" #> CONC file layer range; comment to write all layers to CONC

#> ACONC file species; comment or set to "ALL" to write all species to ACONC
#export AVG_CONC_SPCS="O3 NO CO NO2 ASO4I ASO4J NH3"
export AVG_CONC_SPCS="ALL"
export ACONC_BLEV_ELEV=" 1 1" #> ACONC file layer range; comment to write all layers to ACONC
export AVG_FILE_ENDTIME=N     #> override default beginning ACONC timestamp [ default: N ]

#> Synchronization Time Step and Tolerance Options
export CTM_MAXSYNC=300      #> max sync time step (sec) [ default: 720 ]
export CTM_MINSYNC=60       #> min sync time step (sec) [ default: 60 ]
export SIGMA_SYNC_TOP=0.7   #> top sigma level thru which sync step determined [ default: 0.7 ]
#export ADV_HDIV_LIM=0.95   #> maximum horiz. div. limit for adv step adjust [ default: 0.9 ]
export CTM_ADV_CFL=0.95     #> max CFL [ default: 0.75]
#export RB_ATOL=1.0E-09     #> global ROS3 solver absolute tolerance [ default: 1.0E-07 ]

#> Science Options
export CTM_OCEAN_CHEM=Y     #> Flag for ocean halgoen chemistry and sea spray aerosol emissions [ default: Y ]
export CTM_WB_DUST=N        #> use inline windblown dust emissions [ default: Y ]
export CTM_WBDUST_BELD=BELD3 #> landuse database for identifying dust source regions
                  #>   [ default: UNKNOWN ]; ignore if CTM_WB_DUST = N
export CTM_LTNG_NO=N        #> turn on lightning NOx [ default: N ]
export KZMIN=Y              #> use Min Kz option in edyintb [ default: Y ],
                  #>   otherwise revert to Kz0UT
export CTM_MOSAIC=N         #> landuse specific deposition velocities [ default: N ]
export CTM_FST=N            #> mosaic method to get land-use specific stomatal flux
                  #>   [ default: N ]
export PX_VERSION=Y         #> WRF PX LSM
export CLM_VERSION=N        #> WRF CLM LSM
export NOAH_VERSION=N       #> WRF NOAH LSM
export CTM_ABFLUX=Y         #> ammonia bi-directional flux for in-line deposition
                  #>   velocities [ default: N ]
export CTM_BIDI_FERT_NH3=T  #> subtract fertilizer NH3 from emissions because it will be handled
                  #>   by the BiDi calculation [ default: Y ]
export CTM_HGBIDI=N         #> mercury bi-directional flux for in-line deposition
                  #>   velocities [ default: N ]
export CTM_SFC_HONO=Y       #> surface HONO interaction [ default: Y ]
export CTM_GRAV_SETL=Y      #> vdiff aerosol gravitational sedimentation [ default: Y ]
export CTM_BIOGEMIS_BE=Y    #> calculate in-line biogenic emissions with BEIS
export CTM_BIOGEMIS_MG=N    #> turns on MEGAN biogenic emission [ default: N ]
export BDSNP_MEGAN=N        #> turns on BDSNP soil NO emissions [ default: N ]

#> Vertical Extraction Options
export VERTEXT=N
export VERTEXT_COORD_PATH=${WORKDIR}/lonlat.csv

#> I/O Controls
export IOAPI_LOG_WRITE=F    #> turn on excess WRITE3 logging [ options: T | F ]
export FL_ERR_STOP=N        #> stop on inconsistent input files
export PROMPTFLAG=F         #> turn on I/O-API PROMPT*FILE interactive mode [ options: T | F ]
export IOAPI_OFFSET_64=YES  #> support large timestep records (>2GB/timestep record) [ options: YES | NO ]
export IOAPI_CHECK_HEADERS=N #> check file headers [ options: Y | N ]
```

```
export CTM_EMISCHK=N       #> Abort CMAQ if missing surrogates from emissions Input files
export EMISDIAG=F          #> Print Emission Rates at the output time step after they have been
                  #>  scaled and modified by the user Rules [options: F | T or 2D | 3D | 2DSUM ]
                  #>  Individual streams can be modified using the variables:
                  #>     GR_EMIS_DIAG_## | STK_EMIS_DIAG_## | BIOG_EMIS_DIAG
                  #>     MG_EMIS_DIAG   | LTNG_EMIS_DIAG   | DUST_EMIS_DIAG
                  #>     SEASPRAY_EMIS_DIAG
                  #>  Note that these diagnostics are different than other emissions diagnostic
                  #>  output because they occur after scaling.
export EMISDIAG_SUM=F      #> Print Sum of Emission Rates to Gridded Diagnostic File

#> Diagnostic Output Flags
export CTM_CKSUM=Y         #> checksum report [ default: Y ]
export CLD_DIAG=N          #> cloud diagnostic file [ default: N ]

export CTM_PHOTDIAG=N      #> photolysis diagnostic file [ default: N ]
export NLAYS_PHOTDIAG="1"   #> Number of layers for PHOTDIAG2 and PHOTDIAG3 from
                  #>    Layer 1 to NLAYS_PHOTDIAG  [ default: all layers ]
#export NWAVE_PHOTDIAG="294 303 310 316 333 381 607"  #> Wavelengths written for variables
                               #>  in PHOTDIAG2 and PHOTDIAG3
                               #>  [ default: all wavelengths ]

export CTM_SSEMDIAG=N      #> sea-spray emissions diagnostic file [ default: N ]
export CTM_DUSTEM_DIAG=N    #> windblown dust emissions diagnostic file [ default: N ];
                  #>    Ignore if CTM_WB_DUST = N
export CTM_DEPV_FILE=N     #> deposition velocities diagnostic file [ default: N ]
export VDIFF_DIAG_FILE=N    #> vdiff & possibly aero grav. sedimentation diagnostic file [ default: N ]
export LTNGDIAG=N          #> lightning diagnostic file [ default: N ]
export B3GTS_DIAG=N        #> BEIS mass emissions diagnostic file [ default: N ]
export CTM_WVEL=Y          #> save derived vertical velocity component to conc
                  #>    file [ default: Y ]

# ====================================================================
#> Input Directories and Filenames
# ====================================================================

ICpath=$INPDIR/icbc                  #> initial conditions input directory
BCpath=$INPDIR/icbc                   #> boundary conditions input directory
EMISpath=$INPDIR/emis                    #> gridded emissions input directory
IN_PTpath=$INPDIR/emis                   #> point source emissions input directory
IN_LTpath=$INPDIR/lightning               #> lightning NOx input directory
METpath=$INPDIR/met/mcipv5.4                 #> meteorology input directory
#JVALpath=$INPDIR/jproc                   #> offline photolysis rate table directory
OMIpath=$BLD                        #> ozone column data for the photolysis model
EPICpath=$INPDIR/epic                    #> BELD landuse data for windblown dust model
SZpath=$INPDIR/surface                   #> surf zone file for in-line seaspray emissions

# ====================================================================
#> Begin Loop Through Simulation Days
# ====================================================================
rtarray=""

TODAYG=${START_DATE}
TODAYJ=`date -ud "${START_DATE}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ
START_DAY=${TODAYJ}
```

```
STOP_DAY=`date -ud "${END_DATE}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ
NDAYS=0

while [ $TODAYJ -le $STOP_DAY ]          #>Compare dates in terms of YYYYJJJ
do
  NDAYS=`echo "${NDAYS} + 1" | bc -l`

  #> Retrieve Calendar day Information
  YYYYMMDD=`date -ud "${TODAYG}" +%Y%m%d` #> Convert YYYY-MM-DD to YYYYMMDD
  YYYYMM=`date -ud "${TODAYG}" +%Y%m`     #> Convert YYYY-MM-DD to YYYYMM
  YYMMDD=`date -ud "${TODAYG}" +%y%m%d`   #> Convert YYYY-MM-DD to YYMMDD
  YYYYJJJ=$TODAYJ

  #> Calculate Yesterday's Date
  YESTERDAY=`date -ud "${TODAYG}-1days" +%Y%m%d` #> Convert YYYY-MM-DD to YYYYJJJ
# ======================================================================
#> Set Output String and Propagate Model Configuration Documentation
# ======================================================================
  echo ""
  echo "Set up input and output files for Day ${TODAYG}."

  #> set output file name extensions
  export CTM_APPL=${RUNID}_${YYYYMMDD}

  #> Copy Model Configuration To Output Folder
  if [ ! -d $OUTDIR ]; then
    mkdir -p $OUTDIR
  fi
  cp $BLD/CCTM_${VRSN}.cfg $OUTDIR/CCTM_${CTM_APPL}.cfg

# ======================================================================
#> Input Files (Some are Day-Dependent)
# ======================================================================

  #> Initial conditions
  if [ "$NEW_START" == "true" ] || [ "$NEW_START" == "TRUE" ]
  then
    export ICFILE=CCTM_ICON_v54_${MECH}_12NE3_20180701.nc
    export INIT_MEDC_1=notused
    export INITIAL_RUN=Y #related to restart soil information file
  else
    ICpath=$OUTDIR
    export ICFILE=CCTM_CGRID_${RUNID}_${YESTERDAY}.nc
    export INIT_MEDC_1=$ICpath/CCTM_MEDIA_CONC_${RUNID}_${YESTERDAY}.nc
    export INITIAL_RUN=N
  fi

  #> Boundary conditions
  BCFILE=CCTM_BCON_v54_${MECH}_12NE3_${YYYYMMDD}.nc

  echo "I/BCFILE $ICFILE $BCFILE"
  #> Off-line photolysis rates
  #export JVALfile=JTABLE_${YYYYJJJ}

  #> Ozone column data
```

```
OMIfile=OMI_1979_to_2019.dat

#> Optics file
OPTfile=PHOT_OPTICS.dat

#> MCIP meteorology files
export GRID_BDY_2D=$METpath/GRIDBDY2D_12NE3_${YYYYMMDD}.nc  # GRID files are static, not day-specific
export GRID_CRO_2D=$METpath/GRIDCRO2D_12NE3_${YYYYMMDD}.nc
export GRID_CRO_3D=$METpath/GRIDCRO3D_12NE3_${YYYYMMDD}.nc
export GRID_DOT_2D=$METpath/GRIDDOT2D_12NE3_${YYYYMMDD}.nc
export MET_CRO_2D=$METpath/METCRO2D_12NE3_${YYYYMMDD}.nc
export MET_CRO_3D=$METpath/METCRO3D_12NE3_${YYYYMMDD}.nc
export MET_DOT_3D=$METpath/METDOT3D_12NE3_${YYYYMMDD}.nc
export MET_BDY_3D=$METpath/METBDY3D_12NE3_${YYYYMMDD}.nc
export LUFRAC_CRO=$METpath/LUFRAC_CRO_12NE3_${YYYYMMDD}.nc

#> Control Files
#>
#> IMPORTANT NOTE
#>
#> The DESID control files defined below are an integral part of controlling the behavior of the model simulation.
#> Among other things, they control the mapping of species in the emission files to chemical species in the model
and
#> several aspects related to the simulation of organic aerosols.
#> Please carefully review the DESID control files to ensure that they are configured to be consistent with the
assumptions
#> made when creating the emission files defined below and the desired representation of organic aerosols.
#> For further information, please see:
#> + AERO7 Release Notes section on 'Required emission updates':
#>  https://github.com/USEPA/CMAQ/blob/master/DOCS/Release_Notes/aero7_overview.md
#> + CMAQ User's Guide section 6.9.3 on 'Emission Compatability':
#>
https://github.com/USEPA/CMAQ/blob/master/DOCS/Users_Guide/CMAQ_UG_ch06_model_configuration_option
s.md#6.9.3_Emission_Compatability
#> + Emission Control (DESID) Documentation in the CMAQ User's Guide:
#>
https://github.com/USEPA/CMAQ/blob/master/DOCS/Users_Guide/Appendix/CMAQ_UG_appendixB_emissions_c
ontrol.md
#>
export DESID_CTRL_NML=${BLD}/CMAQ_Control_DESID.nml
export DESID_CHEM_CTRL_NML=${BLD}/CMAQ_Control_DESID_${MECH}.nml

#> The following namelist configures aggregated output (via the Explicit and Lumped
#> Air Quality Model Output (ELMO) Module), domain-wide budget output, and chemical
#> family output.
export MISC_CTRL_NML=${BLD}/CMAQ_Control_Misc.nml

#> The following namelist controls the mapping of meteorological land use types and the NH3 and Hg emission
#> potentials
export STAGECTRL_NML=${BLD}/CMAQ_Control_STAGE.nml

#> Spatial Masks For Emissions Scaling
#export CMAQ_MASKS=$SZpath/OCEAN_${MM}_L3m_MC_CHL_chlor_a_12NE3.nc #> horizontal grid-dependent
ocean file
export CMAQ_MASKS=$INPDIR/GRIDMASK_STATES_12NE3.nc
```

```
#> Gridded Emissions Files
export N_EMIS_GR=2
EMISfile=emis_mole_all_${YYYYMMDD}_12NE3_nobeis_norwc_2018gc_cb6_18j.ncf
export GR_EMIS_001=${EMISpath}/merged_nobeis_norwc/${EMISfile}
export GR_EMIS_LAB_001=GRIDDED_EMIS
export GR_EM_SYM_DATE_001=F # To change default behaviour please see Users Guide for EMIS_SYM_DATE

EMISfile=emis_mole_rwc_${YYYYMMDD}_12NE3_cmaq_cb6ae7_2018gc_cb6_18j.ncf
export GR_EMIS_002=${EMISpath}/rwc/${EMISfile}
export GR_EMIS_LAB_002=GR_RES_FIRES
export GR_EM_SYM_DATE_002=F # To change default behaviour please see Users Guide for EMIS_SYM_DATE

#> In-line point emissions configuration
export N_EMIS_PT=10        #> Number of elevated source groups

STKCASEG=12US1_2018gc_cb6_18j          # Stack Group Version Label
STKCASEE=12US1_cmaq_cb6ae7_2018gc_cb6_18j  # Stack Emission Version Label

# Time-Independent Stack Parameters for Inline Point Sources
export STK_GRPS_001=$IN_PTpath/ptnonipm/stack_groups_ptnonipm_${STKCASEG}.ncf
export STK_GRPS_002=$IN_PTpath/ptegu/stack_groups_ptegu_${STKCASEG}.ncf
export STK_GRPS_003=$IN_PTpath/othpt/stack_groups_othpt_${STKCASEG}.ncf
export STK_GRPS_004=$IN_PTpath/ptagfire/stack_groups_ptagfire_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_005=$IN_PTpath/ptfire-rx/stack_groups_ptfire-rx_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_006=$IN_PTpath/ptfire-wild/stack_groups_ptfire-wild_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_007=$IN_PTpath/ptfire_othna/stack_groups_ptfire_othna_${YYYYMMDD}_${STKCASEG}.ncf
export STK_GRPS_008=$IN_PTpath/pt_oilgas/stack_groups_pt_oilgas_${STKCASEG}.ncf
export STK_GRPS_009=$IN_PTpath/cmv_c3_12/stack_groups_cmv_c3_12_${STKCASEG}.ncf
export STK_GRPS_010=$IN_PTpath/cmv_c1c2_12/stack_groups_cmv_c1c2_12_${STKCASEG}.ncf

# Emission Rates for Inline Point Sources
export STK_EMIS_001=$IN_PTpath/ptnonipm/inln_mole_ptnonipm_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_002=$IN_PTpath/ptegu/inln_mole_ptegu_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_003=$IN_PTpath/othpt/inln_mole_othpt_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_004=$IN_PTpath/ptagfire/inln_mole_ptagfire_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_005=$IN_PTpath/ptfire-rx/inln_mole_ptfire-rx_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_006=$IN_PTpath/ptfire-wild/inln_mole_ptfire-wild_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_007=$IN_PTpath/ptfire_othna/inln_mole_ptfire_othna_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_008=$IN_PTpath/pt_oilgas/inln_mole_pt_oilgas_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_009=$IN_PTpath/cmv_c3_12/inln_mole_cmv_c3_12_${YYYYMMDD}_${STKCASEE}.ncf
export STK_EMIS_010=$IN_PTpath/cmv_c1c2_12/inln_mole_cmv_c1c2_12_${YYYYMMDD}_${STKCASEE}.ncf

# Label Each Emissions Stream
export STK_EMIS_LAB_001=PT_NONEGU
export STK_EMIS_LAB_002=PT_EGU
export STK_EMIS_LAB_003=PT_OTHER
export STK_EMIS_LAB_004=PT_AGFIRES
export STK_EMIS_LAB_005=PT_RXFIRES
export STK_EMIS_LAB_006=PT_WILDFIRES
export STK_EMIS_LAB_007=PT_OTHFIRES
export STK_EMIS_LAB_008=PT_OILGAS
export STK_EMIS_LAB_009=PT_CMV_C3
export STK_EMIS_LAB_010=PT_CMV_C1C2
```

```
# Allow CMAQ to Use Point Source files with dates that do not
# match the internal model date
# To change default behaviour please see Users Guide for EMIS_SYM_DATE
export STK_EM_SYM_DATE_001=F
export STK_EM_SYM_DATE_002=F
export STK_EM_SYM_DATE_003=F
export STK_EM_SYM_DATE_004=F
export STK_EM_SYM_DATE_005=F
export STK_EM_SYM_DATE_006=F
export STK_EM_SYM_DATE_007=F
export STK_EM_SYM_DATE_008=F

#> Lightning NOx configuration
if [ "$CTM_LTNG_NO" == "Y" ]
then
  export LTNGNO="InLine"    #> set LTNGNO to "Inline" to activate in-line calculation

#> In-line lightning NOx options
  export USE_NLDN=Y       #> use hourly NLDN strike file [ default: Y ]
  if [ "$USE_NLDN" == "Y" ]
  then
    export NLDN_STRIKES=${IN_LTpath}/NLDN_12km_60min_${YYYYMMDD}.ioapi
  fi
  export LTNGPARMS_FILE=${IN_LTpath}/LTNG_AllParms_12NE3.nc #> lightning parameter file
fi

#> In-line biogenic emissions configuration
if [ "$CTM_BIOGEMIS_BE" == "Y" ]
then
  IN_BEISpath=${INPDIR}/surface
  export GSPRO=$BLD/gspro_biogenics.txt
  export BEIS_NORM_EMIS=$IN_BEISpath/beis4_beld6_norm_emis.12NE3.nc
  export BEIS_SOILINP=$OUTDIR/CCTM_BSOILOUT_${RUNID}_${YESTERDAY}.nc
                #> Biogenic NO soil input file; ignore if NEW_START = TRUE
fi
if [ "$CTM_BIOGEMIS_MG" == "Y" ]
then
 export MEGAN_SOILINP=$OUTDIR/CCTM_MSOILOUT_${RUNID}_${YESTERDAY}.nc
                #> Biogenic NO soil input file; ignore if INITIAL_RUN = Y
                #>                    ; ignore if IGNORE_SOILINP = Y
    export MEGAN_CTS=$SZpath/megan3.2/CT3_CONUS.ncf
    export MEGAN_EFS=$SZpath/megan3.2/EFMAPS_CONUS.ncf
    export MEGAN_LDF=$SZpath/megan3.2/LDF_CONUS.ncf
    if [ "$BDSNP_MEGAN" == "Y"]
    then
      export BDSNPINP=$OUTDIR/CCTM_BDSNPOUT_${RUNID}_${YESTERDAY}.nc
      export BDSNP_FFILE=$SZpath/megan3.2/FERT_tceq_12km.ncf
      export BDSNP_NFILE=$SZpath/megan3.2/NDEP_tceq_12km.ncf
      export BDSNP_LFILE=$SZpath/megan3.2/LANDTYPE_tceq_12km.ncf
      export BDSNP_AFILE=$SZpath/megan3.2/ARID_tceq_12km.ncf
      export BDSNP_NAFILE=$SZpath/megan3.2/NONARID_tceq_12km.ncf
    fi
fi

#> In-line sea spray emissions configuration
```

```
 export OCEAN_1=$SZpath/OCEAN_${MM}_L3m_MC_CHL_chlor_a_12NE3.nc #> horizontal grid-dependent ocean
file

 #> Bidirectional ammonia configuration
 if [ "$CTM_ABFLUX" == "Y" ]
 then
   export E2C_SOIL=${EPICpath}/2018r1_EPIC0509_12NE3_soil.nc
   export E2C_CHEM=${EPICpath}/2018r1_EPIC0509_12NE3_time${YYYYMMDD}.nc
   export E2C_CHEM_YEST=${EPICpath}/2018r1_EPIC0509_12NE3_time${YESTERDAY}.nc
   export E2C_LU=${EPICpath}/beld4_12NE3_2011.nc
 fi

#> Inline Process Analysis
 export CTM_PROCAN=N       #> use process analysis [ default: N]
#  if ( $?CTM_PROCAN ) then   # $CTM_PROCAN is defined
#    if ( $CTM_PROCAN == 'Y' || $CTM_PROCAN == 'T' ) then
##> process analysis global column, row and layer ranges
##      setenv PA_BCOL_ECOL "10 90"  # default: all columns
##      setenv PA_BROW_EROW "10 80"  # default: all rows
##      setenv PA_BLEV_ELEV "1  4"   # default: all levels
#      setenv PACM_INFILE ${NMLpath}/pa_${MECH}.ctl
#      setenv PACM_REPORT $OUTDIR/"PA_REPORT".${YYYYMMDD}
#    endif
#  endif

#> Sulfur Tracking Model (STM)
 export STM_SO4TRACK=N       #> sulfur tracking [ default: N ]
# if ( $?STM_SO4TRACK ) then
#    if ( $STM_SO4TRACK == 'Y' || $STM_SO4TRACK == 'T' ) then

#     #> option to normalize sulfate tracers [ default: Y ]
#     setenv STM_ADJSO4 Y

#    endif
# endif

# ========================================================================
#> Output Files
# ========================================================================

 #> set output file names
 export S_CGRID="$OUTDIR/CCTM_CGRID_${CTM_APPL}.nc"              #> 3D Inst. Concentrations
 export CTM_CONC_1="$OUTDIR/CCTM_CONC_${CTM_APPL}.nc -v"          #> On-Hour Concentrations
 export A_CONC_1="$OUTDIR/CCTM_ACONC_${CTM_APPL}.nc -v"           #> Hourly Avg. Concentrations
 export MEDIA_CONC="$OUTDIR/CCTM_MEDIA_CONC_${CTM_APPL}.nc -v"      #> NH3 Conc. in Media
 export CTM_DRY_DEP_1="$OUTDIR/CCTM_DRYDEP_${CTM_APPL}.nc -v"       #> Hourly Dry Deposition
 export CTM_DEPV_DIAG="$OUTDIR/CCTM_DEPV_${CTM_APPL}.nc -v"        #> Dry Deposition Velocities
 export B3GTS_S="$OUTDIR/CCTM_B3GTS_S_${CTM_APPL}.nc -v"          #> Biogenic Emissions
 export SOILOUT="$OUTDIR/CCTM_SOILOUT_${CTM_APPL}.nc"             #> Soil Emissions
 export CTM_WET_DEP_1="$OUTDIR/CCTM_WETDEP1_${CTM_APPL}.nc -v"      #> Wet Dep From All Clouds
 export CTM_WET_DEP_2="$OUTDIR/CCTM_WETDEP2_${CTM_APPL}.nc -v"      #> Wet Dep From SubGrid Clouds
 export CTM_PMDIAG_1="$OUTDIR/CCTM_PMDIAG_${CTM_APPL}.nc -v"       #> On-Hour Particle Diagnostics
 export CTM_APMDIAG_1="$OUTDIR/CCTM_APMDIAG_${CTM_APPL}.nc -v"      #> Hourly Avg. Particle Diagnostics
 export CTM_RJ_1="$OUTDIR/CCTM_PHOTDIAG1_${CTM_APPL}.nc -v"        #> 2D Surface Summary from Inline
Photolysis
```

```
 export CTM_RJ_2="$OUTDIR/CCTM_PHOTDIAG2_${CTM_APPL}.nc -v"       #> 3D Photolysis Rates
 export CTM_RJ_3="$OUTDIR/CCTM_PHOTDIAG3_${CTM_APPL}.nc -v"       #> 3D Optical and Radiative Results from
Photolysis
 export CTM_SSEMIS_1="$OUTDIR/CCTM_SSEMIS_${CTM_APPL}.nc -v"      #> Sea Spray Emissions
 export CTM_DUST_EMIS_1="$OUTDIR/CCTM_DUSTEMIS_${CTM_APPL}.nc -v"   #> Dust Emissions
 export CTM_IPR_1="$OUTDIR/CCTM_PA_1_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IPR_2="$OUTDIR/CCTM_PA_2_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IPR_3="$OUTDIR/CCTM_PA_3_${CTM_APPL}.nc -v"          #> Process Analysis
 export CTM_IRR_1="$OUTDIR/CCTM_IRR_1_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_IRR_2="$OUTDIR/CCTM_IRR_2_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_IRR_3="$OUTDIR/CCTM_IRR_3_${CTM_APPL}.nc -v"         #> Chem Process Analysis
 export CTM_DRY_DEP_MOS="$OUTDIR/CCTM_DDMOS_${CTM_APPL}.nc -v"     #> Dry Dep
 export CTM_DRY_DEP_FST="$OUTDIR/CCTM_DDFST_${CTM_APPL}.nc -v"     #> Dry Dep
 export CTM_DEPV_MOS="$OUTDIR/CCTM_DEPVMOS_${CTM_APPL}.nc -v"      #> Dry Dep Velocity
 export CTM_DEPV_FST="$OUTDIR/CCTM_DEPVFST_${CTM_APPL}.nc -v"     #> Dry Dep Velocity
 export CTM_VDIFF_DIAG="$OUTDIR/CCTM_VDIFF_DIAG_${CTM_APPL}.nc -v"  #> Vertical Dispersion Diagnostic
 export CTM_VSED_DIAG="$OUTDIR/CCTM_VSED_DIAG_${CTM_APPL}.nc -v"   #> Particle Grav. Settling Velocity
 export CTM_LTNGDIAG_1="$OUTDIR/CCTM_LTNGHRLY_${CTM_APPL}.nc -v"   #> Hourly Avg Lightning NO
 export CTM_LTNGDIAG_2="$OUTDIR/CCTM_LTNGCOL_${CTM_APPL}.nc -v"    #> Column Total Lightning NO
 export CTM_VEXT_1="$OUTDIR/CCTM_VEXT_${CTM_APPL}.nc -v"         #> On-Hour 3D Concs at select sites

 #> set floor file (neg concs)
 export FLOOR_FILE=${OUTDIR}/FLOOR_${CTM_APPL}.txt

 #> look for existing log files and output files
 ( ls CTM_LOG_???.${CTM_APPL} > buff.txt ) >& /dev/null
 ( ls ${LOGDIR}/CTM_LOG_???.${CTM_APPL} >> buff.txt ) >& /dev/null
 log_test=`cat buff.txt`; rm -f buff.txt

 OUT_FILES=(${FLOOR_FILE} ${S_CGRID} ${CTM_CONC_1} ${A_CONC_1} ${MEDIA_CONC}              \
        ${CTM_DRY_DEP_1} $CTM_DEPV_DIAG $B3GTS_S $SOILOUT $CTM_WET_DEP_1             \
        $CTM_WET_DEP_2 $CTM_PMDIAG_1 $CTM_APMDIAG_1                                  \
        $CTM_RJ_1 $CTM_RJ_2 $CTM_RJ_3 $CTM_SSEMIS_1 $CTM_DUST_EMIS_1 $CTM_IPR_1 $CTM_IPR_2   \
        $CTM_IPR_3 $CTM_IRR_1 $CTM_IRR_2 $CTM_IRR_3 $CTM_DRY_DEP_MOS                 \
        $CTM_DRY_DEP_FST $CTM_DEPV_MOS $CTM_DEPV_FST $CTM_VDIFF_DIAG $CTM_VSED_DIAG       \
        $CTM_LTNGDIAG_1 $CTM_LTNGDIAG_2 $CTM_VEXT_1 )

 OUT_FILES=`echo $OUT_FILES | sed "s; -v;;g" | sed "s;MPI:;;g" `
 ( ls $OUT_FILES > buff.txt ) >& /dev/null
 out_test=`cat buff.txt`; rm -f buff.txt

 #> delete previous output if requested
 if [ "$CLOBBER_DATA" == "true" ] || [ "$CLOBBER_DATA" == "TRUE" ]
 then
   echo
   echo "Existing Logs and Output Files for Day ${TODAYG} Will Be Deleted"

   #> remove previous log files
   for file in $log_test ; do
#      #echo "Deleting log file: $file"
     /bin/rm -f "$file"
   done
   #> remove previous output files
   for file in $out_test ; do
#      #echo "Deleting output file: $file"
```

```
    /bin/rm -f "$file"
  done
  /bin/rm -f ${OUTDIR}/CCTM_EMDIAG*${RUNID}_${YYYYMMDD}.nc

else
  #> error if previous log files exist
  if [ "$log_test" -ne "" ]
  then
    echo "*** Logs exist - run ABORTED ***"
    echo "*** To overide, set CLOBBER_DATA = TRUE in run_cctm.csh ***"
    echo "*** and these files will be automatically deleted. ***"
    exit 1
 fi

  #> error if previous output files exist
  if [ "$out_test" != "" ]
  then
    echo "*** Output Files Exist - run will be ABORTED ***"
  for file in $out_test ; do
     echo " cannot delete $file"
  done
    echo "*** To overide, set CLOBBER_DATA = TRUE in run_cctm.csh ***"
    echo "*** and these files will be automatically deleted. ***"
    exit 1
  fi
fi

 #> for the run control ...
 export CTM_STDATE=$YYYYJJJ
 export CTM_STTIME=$STTIME
 export CTM_RUNLEN=$NSTEPS
 export CTM_TSTEP=$TSTEP
 export INIT_CONC_1=$ICpath/$ICFILE
 export BNDY_CONC_1=$BCpath/$BCFILE
 export OMI=$OMIpath/$OMIfile
 export OPTICS_DATA=$OMIpath/$OPTfile
#export XJ_DATA=$JVALpath/$JVALfile

 #> species defn & photolysis
 export gc_matrix_nml=${NMLpath}/GC_$MECH.nml
 export ae_matrix_nml=${NMLpath}/AE_$MECH.nml
 export nr_matrix_nml=${NMLpath}/NR_$MECH.nml
 export tr_matrix_nml=${NMLpath}/Species_Table_TR_0.nml

 #> check for photolysis input data
 export CSQY_DATA=${NMLpath}/CSQY_DATA_$MECH


#|#  if [ ! (-e $CSQY_DATA ) ]
#|#  then
#|#    echo " $CSQY_DATA  not found "
#|#    exit 1
#|#  fi
#|#  if [ ! (-e $OPTICS_DATA ) ]
#|#  then
```

```
#|#    echo " $OPTICS_DATA  not found "
#|#    exit 1
#|#  fi

# ========================================================================
#> Execution Portion
# ========================================================================

 #> Print attributes of the executable
#|#  if [ $CTM_DIAG_LVL != 0 ]
#|#  then
#|#    ls -l $BLD/$EXEC
#|#    size $BLD/$EXEC
#|#    unlimit
#|#    limit
#|#  fi

 #> Print Startup Dialogue Information to Standard Out
 echo
 echo "CMAQ Processing of Day $YYYYMMDD Began at `date`"
 echo

 #> Executable call for single PE, uncomment to invoke
 #( /usr/bin/time -p $BLD/$EXEC ) |& tee buff_${EXECUTION_ID}.txt

 #> Executable call for multi PE, configure for your system
 # MPI=/usr/local/intel/impi/3.2.2.006/bin64
 # MPIRUN=$MPI/mpirun
 ( /usr/bin/time -p mpirun -np $NPROCS $VEXILLA $BLD/$EXEC ) |& tee buff_${EXECUTION_ID}.txt

 #> Harvest Timing Output so that it may be reported below
 rtarray="${rtarray} `tail -3 buff_${EXECUTION_ID}.txt | grep -Eo '[+-]?[0-9]+([.][0-9]+)?' | head -1` "
 rm -rf buff_${EXECUTION_ID}.txt

 #> Abort script if abnormal termination
 if [ ! -e $OUTDIR/CCTM_CGRID_${CTM_APPL}.nc ]
 then
  echo ""
  echo "**************************************************************"
  echo "** Runscript Detected an Error: CGRID file was not written. **"
  echo "**   This indicates that CMAQ was interrupted or an issue   **"
  echo "**   exists with writing output. The runscript will now     **"
  echo "**   abort rather than proceeding to subsequent days.       **"
  echo "**************************************************************"
  break
 fi

 #> Print Concluding Text
 echo
 echo "CMAQ Processing of Day $YYYYMMDD Finished at `date`"
 echo
 echo "\\\\\=====\\\\\=====\\\\\=====\\\\\=====/////=====/////=====/////=====/////"
 echo

# ========================================================================
```

```
#> Finalize Run for This Day and Loop to Next Day
# ======================================================================

 #> Save Log Files and Move on to Next Simulation Day
 echo "CTM_LOG_???.${CTM_APPL} and LOGDIR $LOGDIR"
 mv CTM_LOG_???.${CTM_APPL} $LOGDIR
 if [ $CTM_DIAG_LVL != 0 ]
 then
  mv CTM_DIAG_???.${CTM_APPL} $LOGDIR
 fi

 #> The next simulation day will, by definition, be a restart
 export NEW_START=false

 #> Increment both Gregorian and Julian Days
 TODAYG=`date -ud "${TODAYG}+1days" +%Y-%m-%d` #> Add a day for tomorrow
 TODAYJ=`date -ud "${TODAYG}" +%Y%j` #> Convert YYYY-MM-DD to YYYYJJJ

done
#|#done  #> Loop to the next Simulation Day

# ======================================================================
#> Generate Timing Report
# ======================================================================
RTMTOT=0
for it in `seq ${NDAYS}` ; do
   rt=`echo ${rtarray} | cut -d' ' -f${it}`
   RTMTOT=`echo "${RTMTOT} + ${rt}" | bc -l`
done

RTMAVG=`echo "scale=2; ${RTMTOT} / ${NDAYS}" | bc -l`
RTMTOT=`echo "scale=2; ${RTMTOT} / 1" | bc -l`

echo
echo "================================="
echo "  ***** CMAQ TIMING REPORT *****"
echo "================================="
echo "Start Day: ${START_DATE}"
echo "End Day:   ${END_DATE}"
echo "Number of Simulation Days: ${NDAYS}"
echo "Domain Name:             ${GRID_NAME}"
echo "Number of Grid Cells:     ${NCELLS}  (ROW x COL x LAY)"
echo "Number of Layers:        ${NZ}"
echo "Number of Processes:      ${NPROCS}"
echo "   All times are in seconds."
echo
echo "Num  Day       Wall Time"
d=0
day=${START_DATE}

for it in `seq ${NDAYS}` ; do
   # Set the right day and format it
   d=`echo "${d} + 1"  | bc -l`
   n=`printf "%02d" ${d}`
```

```
   # Choose the correct time variables
   rt=`echo ${rtarray} | cut -d' ' -f${it}`

   # Write out row of timing data
   echo "${n}   ${day}   ${rt}"

   # Increment day for next loop
   day=`date -ud "${day}+1days" +%Y-%m-%d`
done

echo "     Total Time = ${RTMTOT}"
echo "      Avg. Time = ${RTMAVG}"

exit
```