I.2-UNIX-CALB PROGRAM EXECUTION INFORMATION FOR CALIBRATION SYSTEM PROGRAMS ON UNIX SYSTEMS

Bookmarks

Contents

[Next] [Previous] [Bookmarks] [Top]

Script

The script calb can be used to execute the following Calibration System programs:

- gs2oh
- map
- mape
- mapx
- mcp3
- mat
- opt3
- pxpp

- taplot
- ts2oh

The command format is:

```
calb -p progname
     [-d]
     [-i in_file] [-o out_file_prefix]
     [-u user] [-g input_group] [-G data_group]
     [-m] [-t] [-x]
```

The only required parameters are the program to be executed specified by the -p option and the input file name specified by the -i option.

All other parameters are optional and are set to default values if not entered on the command line.

Available options are:

Option	Description	Default Value
-c	Other command line arguments that a command may use; put in quotes if more than one	No arguments
-d	Use development executable directory specified by the token my_rls	Use directory specified by the token calb_rls
-g	Input file group override	The group specified by the token calb_inpt_grp
-G	Data file group override	The group specified by the token calb_data_grp
-i	Input file name	None
-m	Do not print script information	Print messages messages
-o	Output file prefix or 'tty'; output files are date-time stamped and are placed in the user's output directory; if 'tty' is specified output is written to the terminal	progname
-p	Program name	None
-t	Output log information to the	Output to log file terminal
-u	User name override; used to place output in output directory other than	Login user name ($LOGNAME)

Option	Description	Default Value
	the submitting user's	
-x	Conduct execution check only; display additional information but do not execute program	Program is executed

[Next] [Previous] [Bookmarks] [Top]

Apps Default Tokens

The Calibration System scripts and programs use the Apps Defaults System to set execution options and path names (see Chapter I.2-UNIX-APPSDFLT [Hyperlink]).

The following is a description of some of the tokens used:

Token	Description
calb_dir	pathname for calb system level (e.g. /apps/nwsrfs/calb)
calb_data_grp	name of calb data group (e.g. nwrfc, test, <user>)
calb_inpt_grp	name of calb input group (e.g. abrfc, test)
calb_input	pathname for calb input by program including the calb input group; default is '$(calb_dir)/input/$(calb_inpt_grp)'; can use '$(calb_dir)/input' if no calb_inpt_grp level is specified
calb_output	pathname for calb output user directories and does not include the user directory name; default is '$(calb_dir)/output'
calb_sta_ts_dir	pathname for calb station time series data directory; default is '$(calb_dir)/data/sta_ts/$(calb_data_grp)'; can use '$(calb_dir)/data/sta_ts' if no calb_data_grp level is specified
calb_area_ts_dir	pathname for calb area time series data directory ; default is '$(calb_dir)/data/area_ts/$(calb_data_grp)'; can use '$(calb_dir)/data/area_ts' if no calb_data_grp level is specified
calb_input	is $(calb_dir)/input/$(calb_inpt_grp)
calb_dir	is $(apps_dir)/nwsrfs/calb
calb_rls	is $(calb_dir)/bin/RELEASE

Token	Description
calb_output	is $(calb_dir)/output

[Next] [Previous] [Bookmarks] [Top]

Directory Structure

Data:

There is a directory under the ../nwsrfs/calb directory called data. This will have data for all the Calibration System programs. It has subdirectories called sta_ts and area_ts.

The sta_ts directory is for the station time series used as input by the Calibration System programs. These time series would be the output from the Internet programs used to download data or the Historical Data Browser program. Those programs do not automatically put the time series files into this directory structure so they may have to be moved to the sta_ts directories after obtaining the station data. The sta_ts directory can have subdirectories. A particular subdirectory is referred to by a apps_default token calb_data_grp. The calb_data_grp could represent a region, a user, an RFC, etc. Each of the subdirectories in sta_ts should have evap, pcpn and tempt subdirectories. These subdirectories will hold the station time series for the MAPE (evap), MAP (pcpn), PXPP (pcpn), MAT (tempt) and TAPLOT (tempt) programs. A possible directory structure would be:

```
.../calb/data/sta_ts/region/evap/
                           /pcpn/
                           /tempt/
```

where region is the calb_data_grp.

The area_ts subdirectory of data will contain all the time series needed to run mcp3 for a basin. The term basin is equivalent to a segment in OFS and is the smallest division that mcp3 would be run for. Area_ts can have subdirectories. A particular subdirectory is referred to by the apps_default token calb_data_grp. The programs will write their output time series to this part of the data directory tree. The two levels of subdirectories below the calb_data_grp are obtained from the program input decks. These directories must exist for the programs to be able to write there.

Data needed for mcp3 for a basin but not derived from a program, such as QME data obtained Internet programs used to download data or the Historical Data Browser, should also be put into this directory structure. A possible directory structure could look like the following:

```
.../calb/data/area_ts/region/fgroup1/basin1/
                                    /basin2/
                            /fgroup2/basin1/
                                    /basin2/
```

where region is the calb_data_grp

Input:

The ../nwsrfs/calb/input directory can have subdirectories. A particular subdirectory is referred to by the apps_default token calb_inpt_grp. This is similar to the ofs_level apps_default token ofs_level used to determine which set of input files to look at. Under each input group should be subdirectories for each of the Calibration System programs. The input files should be under the subdirectory with the program name. For example an map input file would be placed under the directory

```
.../nwsrfs/calb/input/hrl/map
```

where hrl is the calb_inpt_grp

If the input file is not found in the ../nwsrfs/calb/input directory then the current directory is checked. If it is found in the current directory then the output may be written to the current directory.

Output:

The non-time-series output from the runs (run summary files, log files, punch files and input files for the IDMA program) are written to a subdirectory of the directory the apps_default token calb_output points to. That subdirectory is the one with the $LOGNAME (the user's UNIX identifier). This is similar to the apps_default token ofs_output used to determine where the output files are to be written. The subdirectory must exist for the programs to work correctly. For example if calb_output points to

```
.../nwsrfs/calb/output
```

and user Joe did the run then the output would be found in

```
.../nwsrfs/calb/output/Joe
```

If the input file specified by the -i option is found in the current directory then the output may be written to the current directory.

Files:

The following is a description of the file structure and input/output for the Calibration System programs.

File structure:

```
calb /data /sta_ts  /<cdgrp>/pcpn /<px-fls>
                             tempt/<tm-fls>
                             evap /<ev-fls>
            area_ts /<cdgrp>/<fcgrp>/<basn>/<bn-fls>
                             <basn>/<bn-fls>
      input/<cigrp>/mpc3  /<in-fls>
```

```
                   opt3  /<in-fls>
                   map   /<in-fls>
                   mape  /<in-fls>
                   mat   /<in-fls>
                   pxpp  /<in-fls>
                   taplot/<in-fls>
       output/<user><ot-fls>
                 <pu-fls>
                 <lg-fls>

  where <cdgrp>   is from token calb_data_grp and describes a region
                    or user level; or it could be 'test' for a set of
                    test data; or it could be $LOGNAME for a
                    particular user; it applies to both the station
                    and area time series levels
        <cigrp>   is from token calb_inpt_grp and describes a region
                    or user level as above but is allowed to be
                    different than <cdgrp>
        <user>    is the user level for the output files and comes
                    from either the global variable $LOGNAME or from
                    a script supplied argument
        <fcgrp>   is the Forecast Group directory that is part of the
                    input data to the programs- this directory must
                    be exist before the program are run
        <basn>    is the basin directory that is part of the input
                    data to the programs; these directories must be
                    exist before the program are run
        <px-fls>  are the precipitation files used as input files in
                    programs PXPP or MAP
        <tm-fls>  are the temperature files used as input files in
                    programs MAT or TAPLOT
        <ev-fls>  are the evaporation files used as input files in
                    program MAPE
        <bn-fls>  are the basin area time series files produced by
                    programs MAT, MAP and MAPE; output file names
                    contain suffixes such as '<basn>.MAT',
                    '<basn>.MAP06', '<basn>.MAP24', etc.
        <in-fls>  are the input files in card image format for the
                    programs
        <ot-fls>  are the output files summarizing the program runs
        <pu-fls>  are the output punch files
        <lg-fls>  are the log files; the filename is created by the
                    program scripts by appending '_log' and a
                    date-time stamp to the program name; the
                    following apps_defaults tokens control the output
                    to this file:
                    ofs_log_output=on     output file open/close
                                          statements
                    ofs_error_output=on   output input/output error
                                          statements
```

[Next] [Previous] [Bookmarks] [Top]

File Names

The complete file name is generated by appending the file name provided in the input deck to the directory name provided with the calb_area_ts_dir token. Therefore, the combination of the input filename and the calb_area_ts_dir token value must provide the complete pathname and filename to the data file.

For example if the complete path and filename is

```
   .../nwsrfs/calb/input/mcp3/data/alabama/tlng1/tlng1.map
```

then calb_area_ts_dir may be set to

```
   calb_area_ts_dir : .../nwsrfs/calb/input/mcp3/data/alabama/tlng1
```

and the filename in the input deck would be

```
   tlng1.map
```

or the calb_area_ts_dir could be set to

```
   calb_area_ts_dir : .../nwsrfs/calb/input/mcp3/data/alabama
```

and the filename in the input deck would be

```
   tlng1/tlng1.map
```

The programs provides the '/' between the directory and the pathname.

[Next] [Previous] [Bookmarks] [Top]

Temporary Files

Programs MAP, MAPE, MAT and MCP3 use temporary files for data storage.

On the UNIX system they are hidden files and can be seen using the command 'ls -a'.

The location of the file can be controlled using the TMPDIR environment variable.

The following rules determines where the temporary file will be located:

- the directory specified by the TMPDIR environment variable if it is defined in the user's environment or
- the directory specified by the P_tmpdir variable in /usr/include/stdio.h (/var/tmp on the AWIPS) or
- /tmp

For program MAP, the size of the file depends on the number of stations and on the number of months and can be calculated as follows:

```
   bytes = (num_stations)*(num_months)*((745*4)+4)
   where bytes         is the file size in bytes
         num_stations  is the number of stations
         num_months    is the number of months in the period of record
```

This file can be quite large and users should check for sufficient space in the directory in which it is to be created before running.

[Next] [Previous] [Bookmarks] [Top]

Programs gs2oh and ts2oh

The pathnames that will be used can be obtained without running the programs by using the '-x' option. For example:

```
   calb -x -p gs2oh -i <in> -o <out>
```

[Back] [Next] [Previous] [Bookmarks] [Top]

gs2oh

```
   calb  -p gs2oh  -i <input_name>  -o <output_name>
```

Input file:

The <input_name> must be given. The pathname to the input file is made using apps_defaults token 'calb_input'.

If the input file does not exist using the current directory is searched for the file.

Output file:

The '-o <output_name>' argument is optional.

If not given then the output name is set to 'gs2oh_output'.

If the input file was obtained using the apps_defaults token calb_input then the output directory will use apps_defaults token calb_output/$LOGNAME to create the full pathname of the output file. If the option '-u <user>' is given then $LOGNAME is replaced by the given <user> directory name.

If the user subdirectory under 'calb_output' is missing and $LOGNAME is used then the subdirectory is created.

A date-time stamp in the form YYYYMMDD.HHMMSS is added to the end of the output file.

If the input file was obtained from the current directory (meaning that file calb_input/input_name does not exist) then the output file is made in the current directory. It will not have a date-time stamp added to it.

Do not use an output filename of 'tty'. Unlike most other calb programs using 'tty' for gs2oh will not send output to standard out.

Log file:

A log file is created in the directory calb_output/$LOGNAME which must exist. 'LOGNAME' may be over-ridden by using the option '-u <user>'. The log file has a date-time stamp appended to it.

Other command line options:

Optional command line arguments for gs2oh can be given with option '-c "<other args>"'. Quotes around the arguments are needed if there is more than one argument.

Argument options for gs2oh are:

```
    -m     ....... convert English units to Metric units; default is
                      English units
    -Fn.n  ....... output values in the specified Fortran format;
                      default is F9.3; n must be less than 10
```

Apps Default Tokens:

The number of columns to be output in the DATACARD format [Hyperlink] file (1, 4 or 6) is set to a default value of 4 in the calb script. The value can be changed by exporting the variable 'GS2OH_FIELDS' before running the script.

[Next] [Previous] [Bookmarks] [Top]

ts2oh

```
calb  -p ts2oh  -i <input_name>  -o <output_name>
```

Input file:

The <input_name> must be given. The pathname to the input file will then be made using apps_defaults token 'calb_input'.

If the input file does not exist using these tokens then the given input file is searched for in the current directory.

All the rules for the input file are the same as for program gs2oh [Bookmark].

Output file:

The output is placed in a subdirectory 'outdir'. If the input file comes from the current directory then the subdirectory 'outdir' is created in the current directory; but if the input comes from using apps_defaults token 'calb_input' then the subdirectory 'outdir' is created in the directory found by using apps_defaults token 'calb_output'.

The name of the output files are created by the program and are like:

```
SALTRT.SQIN.19960301.19960407.datacard
```

The '-o <output_name>' argument only affects the name of the output log filename.

Log file:

A log file is created in the output directory defined by 'calb_output'/$LOGNAME even if the input and output are from the current directory. If argument '-o <output_name>' is given then the name 'output_name' is used with '_log' and a date-time stamp appended to it; if it is not given then the name 'ts2oh_output' followed by '_log' and a date-time stamp is used.

Creating merged DATACARD files:

Program TS2OH can read an existing DATACARD file, merge the data with a TSDATA output and write updated DATACARD file.

The following example shows how 3 files would be converted and merged:

```
calb -p ts2oh -i file1.txt     # first file
mv ./outdir/*.datacard .       # move output file so it can be
                               used as input
calb -p ts2oh -i file2.txt     # second file
rm *.datacard                  # remove input DATACARD file just
                               used
mv ./outdir/*.datacard .       # move output file so it can be
                               used as input
calb -p ts2oh -i file3.txt     # third file
rm *.datacard                  # remove input DATACARD file just
                               used
mv ./outdir/*.datacard .       # move output file
```

[Next] [Previous] [Bookmarks] [Top]